Engineering Applications of Artificial Intelligence 49 (2016) 149–166

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

A structured argumentation system with backing and undercutting

Andrea Cohen *, Alejandro J. García, Guillermo R. Simari

Institute for Research in Computer Science and Engineering (ICIC), Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Universidad Nacional del Sur,
Av. Alem 1253 – B8000CPB Bahía Blanca – Argentina

ARTICLE INFO

Article history:
Received 30 December 2014
Received in revised form
28 September 2015
Accepted 1 October 2015
Available online 30 October 2015

Keywords:
Argumentation
Structured argumentation
Logic programming
Non-monotonic reasoning
Knowledge representation
Attack and support for inferences

ABSTRACT

This work introduces *Extended Defeasible Logic Programming* (*E-DeLP*), a structured argumentation system enabling the expression of reasons for and against using defeasible rules. E-DeLP extends the formalism of Defeasible Logic Programming (DeLP) by incorporating two new kinds of rules: backing and undercutting rules. In that way, E-DeLP accounts for Toulmin's notion of backing and Pollock's notion of undercut, two important contributions within the field of argumentation. Thus, E-DeLP constitutes a novel approach, being the first structured argumentation system in providing a unified setting for modeling Toulmin's and Pollock's ideas simultaneously.

© 2015 Elsevier Ltd. All rights reserved.

1. Introduction

Argumentation is a way of reaching conclusions where explicit attention is given to the reasons for and against a particular belief. In this kind of reasoning, a claim is accepted or rejected according to the analysis of the arguments in favor and contrary to it. The way in which arguments and justifications for a claim are considered allows for an automatic reasoning mechanism from a knowledge base that could contain contradictory, incomplete, and uncertain information.

In the last decades, the computational form of argumentation has evolved as an attractive paradigm for conceptualizing commonsense reasoning (Prakken and Vreeswijk, 2002; Bench-Capon and Dunne, 2007; Besnard and Hunter, 2008; Rahwan and Simari, 2009). Nowadays, there exist a variety of practical approaches to model different aspects of argumentation such as abstract argumentation frameworks (Dung, 1995) and structured argumentation systems (Reed et al., 2014), making this kind of reasoning particularly attractive for decision making problems. Moreover, since within the area of argumentation in Artificial Intelligence there exists keen interest in effectively addressing that kind of problems, concrete applications of argumentation systems have appeared in domains like legal reasoning (Prakken and Sartor, 2002; Verheij, 2003), agent dialogues and negotiation (Amgoud et al., 2000; Black and Hunter, 2009), and multi-agent systems (Parsons et al., 1998; Amgoud et al., 2002).

A usual critique to the existing structured argumentation systems is that certain reasoning patterns studied in areas like legal reasoning and philosophy, which constitute important contributions to the argumentation community, were simplified or not considered in their formalization; such is the case of the contributions by Toulmin (1958) and Pollock (1987). In his stimulating work, Toulmin promoted the idea that arguments needed to be analyzed using a richer formalism than the duality of premises and conclusion used in formal logic. Thus, he proposed a model for the layout of arguments that, in addition to *data* and *claim*, distinguishes other components, namely *warrant*, *backing*, *rebuttal* and *qualifier*. In particular, the warrant conveys an inferential connection between the premises (data) and conclusion (claim) of the argument, and the backing establishes the reasons why the warrant holds. Therefore, in addition to the data supporting the claim, the backing provides support for the inference modeled by the warrant of the argument. The foundational work of Pollock introduced a classification of possible defeaters for an argument, distinguishing between rebutting defeaters and undercutting defeaters. Whereas the former dispute the conclusion of an argument, the latter challenge the connection between premises and conclusion of an argument (*i.e.,* its associated warrant in Toulmin's terminology).

The work by Toulmin suggests that the notion of support was present since the foundational works in argumentation. However, following the seminal work on abstract argumentation by Dung

* Corresponding author.
E-mail addresses: ac@cs.uns.edu.ar (A. Cohen), ajg@cs.uns.edu.ar (A.J. García), grs@cs.uns.edu.ar (G.R. Simari).

http://dx.doi.org/10.1016/j.engappai.2015.10.001
0952-1976/© 2015 Elsevier Ltd. All rights reserved.

(1995), later studies on argumentation put aside the notion of support to focus on the notion of attack. Notwithstanding this, in the last decade, the study of the notion of support regained attention amongst researchers in the area of abstract argumentation, considering alternative interpretations of support through the formalization of a general support relation (Cayrol and Lagasquie-Schiex, 2005), an evidential support relation (Oren and Norman, 2008), a deductive support relation (Boella et al., 2010), a necessary support relation (Nouioua and Risch, 2011) and a backing relation (Cohen et al., 2012).

Regarding Toulmin's ideas, Verheij (2005) provided a reconstruction of Toulmin's model in DefLog (Verheij, 2003), thus becoming the first researcher to explicitly address the notion of support in the context of structured argumentation. In addition to Verheij's reconstruction of Toulmin's ideas, other structured argumentation approaches like ASPIC+ (Prakken, 2009) have accounted for Pollock's ideas by allowing for the existence of undercutting attacks. However, none of the existing structured argumentation systems so far have considered Toulmin's and Pollock's ideas simultaneously.

The contribution of this work is to incorporate Toulmin's notion of backing and Pollock's notion of undercut into a structured argumentation system. The formalism proposed in this paper, called *Extended Defeasible Logic Programming* (*E-DeLP*), extends the system of García and Simari (2004) by adding two new types of rules enabling the expression of reasons for and against using defeasible rules. Thus, in E-DeLP it will be possible to build backing and undercutting arguments, hence capturing both Toulmin's and Pollock's ideas. Our work builds upon Cohen et al. (2011), where an initial version of the E-DeLP formalism was presented. In particular, this work substantially develops the ideas presented in Cohen et al. (2011), improving definitions, adding more examples and intuitive explanations, among other changes. Moreover, here we explicitly address the acceptability calculus of arguments in order to determine the inferences of the system, and we identify a series of properties satisfied by it. For that purpose, we will use the abstract argumentation framework of Cohen et al. (2012), which accounts for Toulmin's notion of backing and Pollock's notion of undercut. Hence, we will show that E-DeLP provides the means for instantiating the abstract framework of Cohen et al. (2012).

The rest of this paper is organized as follows. Section 2 presents the representational language of E-DeLP and introduces the syntax of E-DeLP programs. Given the specification of E-DeLP programs, in Section 3 we present a mechanism for building arguments, and we identify three types of arguments: claim arguments, backing arguments, and undercutting arguments. In Section 4 we determine the situations in which arguments conflict with one another, distinguishing between rebutting attacks, undercutting attacks, and undermining attacks. Given the arguments built from an E-DeLP program and the attacks between them, Section 5 provides a way for obtaining the defeats between those arguments, which involves the characterization of a Backing-Undercutting Argumentation Framework (BUAF) (Cohen et al., 2012) associated with the E-DeLP program. Section 6 provides a way of obtaining the acceptability status of arguments in E-DeLP, in terms of their corresponding acceptability status in the associated BUAF. Then, we present a series of properties satisfied by E-DeLP, some of which relate to desired features of argumentation systems. To put this proposal in the research context, Section 7 discusses related work by others, as well as the previous version of E-DeLP proposed in Cohen et al. (2011). Finally, Section 8 presents some conclusions and future lines of work.

2. The representational language

In this section we will present the representational language of Extended Defeasible Logic Programming (E-DeLP). This formalism extends the representational language of DeLP, as proposed in García and Simari (2004). Thus, using this extended language, E-DeLP will allow for the representation of Pollock's notion of undercut and Toulmin's notion of backing.

The representational language of E-DeLP is defined in terms of five disjoint sets: a set of *facts*, a set of *strict rules*, a set of *defeasible rules*, a set of *backing rules*, and a set of *undercutting rules*. These facts and rules are obtained using ground literals, which correspond to ground atoms or negated ground atoms using *strong negation*. As defined in García and Simari (2004) (following the ideas of Alferes et al., 1996), the symbol "$\sim$" represents the strong negation, which differs from the *default negation* frequently used in logic programming and the *classical negation* used in classical logic. The aim of having strong negation in E-DeLP is to allow for the defeasible derivation of potentially contradictory information. As shown below, facts, strict rules, and defeasible rules in E-DeLP are defined in the same way as in DeLP, following the usual convention adopted in logic programming (Lifschitz, 1996).

Definition 1 (*Ground literal*). Let $\mathcal{L}$ be a set of ground atoms. A *ground literal* L (or *literal*, for short) is an atom A or a negated atom $\sim A$, where $A \in \mathcal{L}$.

Definition 2 (*Fact*). A *fact* is a ground literal L.

Definition 3 (*Strict rule*). A *strict rule* is an ordered pair, denoted "*Head* $\leftarrow$ *Body*", where the first element (Head) is a ground literal and the second element (Body) is a finite and non-empty set of ground literals.

A strict rule with head L_0 and body $\{L_1, \ldots, L_n\}$ $(n > 0)$ will also be written as $L_0 \leftarrow L_1, \ldots, L_n$.

Definition 4 (*Defeasible Rule*). A *defeasible rule* is an ordered pair, denoted "*Head* $-\!\prec$ *Body*", where the first element (Head) is a ground literal and the second element (Body) is a finite set of ground literals.

A defeasible rule with head L_0 and body $\{L_1, \ldots, L_n\}$ $(n \geq 0)$ will also be written as $L_0 -\!\prec L_1, \ldots, L_n$.

Facts and strict rules express non-defeasible or indisputable information, whereas defeasible rules express tentative information that may be used if nothing could be posed against it. In addition, a defeasible rule with an empty body is noted as "*Head* $-\!\prec$" and is called a *presumption* (Nute, 1988). Hence, a presumption "*P* $-\!\prec$" will express that "*there are defeasible reasons to believe in P*". The new elements incorporated into the representational language of E-DeLP are the *backing rules* and the *undercutting rules* which, respectively, are used to express reasons for and against defeasible rules. As a result, the addition of these rules enables one to argue about the application of defeasible rules by supporting or attacking them.

Definition 5 (*Backing Rule*). A *backing rule* is an ordered pair, denoted "[*Head*]$\leftarrow\!\!\oplus$[*Body*]", where the first element (*Head*) is a defeasible rule of the form $L_0 -\!\prec L_1, \ldots, L_n$ $(n \geq 0)$ and the second element (*Body*) is a finite and non-empty set of ground literals $\{B_1, \ldots, B_m\}$ $(m > 0)$. A backing rule will also be written as $[L_0 -\!\prec L_1, \ldots, L_n] \leftarrow\!\!\oplus [B_1, \ldots, B_m]$.

Definition 6 (*Undercutting Rule*). An *undercutting rule* is an ordered pair, denoted "[*Head*]$\leftarrow\!\!\otimes$[*Body*]", where the first element (*Head*) is a defeasible rule of the form $L_0 -\!\prec L_1, \ldots, L_n$ $(n \geq 0)$ and the second element (*Body*) is a finite and non-empty set of ground literals $\{U_1, \ldots, U_m\}$ $(m > 0)$. An undercutting rule will also be written as $[L_0 -\!\prec L_1, \ldots, L_n] \leftarrow\!\!\otimes [U_1, \ldots, U_m]$.

Syntactically, the only difference between backing and undercutting rules is the use of the symbols '$\oplus$' and '$\otimes$'. However, the semantics associated with these two kinds of rules is completely

opposite. A backing rule "[*Head*]⟵⊕[*Body*]" expresses that "*reasons to believe in Body provide reasons in favor of using the defeasible rule Head*"; the intuition here is that a backing rule establishes conditions under which the use of the defeasible rule it supports makes sense. An undercutting rule "[*Head*]⟵⊗[*Body*]" expresses that "*reasons to believe in Body provide reasons against using the defeasible rule Head*"; in this case, the intended meaning is that an undercutting rule establishes conditions under which the defeasible rule it is related to should not be used. Both notions are illustrated by the following example.

Example 1. Let us consider an example proposed by Toulmin, illustrated in Fig. 1, which discusses the nationality of a man called Harry. It is possible to represent the different elements in Toulmin's example by using the representational language of E-DeLP. Given the set of atoms $\mathcal{L}_1 =$ {*harry_british*, *harry_born_in_bermuda*, *british_parliament_acts*, *harry_alien_parents*}, the data that Harry was born in Bermuda can be represented by the literal "*harry_born_in_bermuda*", and the claim that Harry is a British subject can be represented by the literal "*harry_british*". Recall that the warrant constitutes an inference rule that provides a connection between data and claim. Thus, the warrant that a man born in Bermuda will generally be a British subject can be represented by the defeasible rule "*harry_british*—<*harry_born_in_bermuda*", since reasons to believe that Harry was born in Bermuda provide reasons to believe that he is British. In particular, the qualifier 'presumably' is not explicitly stated, but it is assumed to be implicit in the defeasibility of the rule modeling the warrant. Also, the backing that establishes the existence of statutes and other legal provisions can be represented by the literal "*british_parliament_acts*". Thus, the support the backing provides for the warrant can be represented by the backing rule:

[*harry_british*—<*harry_born_in_bermuda*]⟵⊕[*british_parliament_acts*]

Finally, the fact that Harry's parents are alien can be represented by the literal "*harry_alien_parents*". This information provides a condition of exception for the warrant and therefore, it constitutes a reason against using the corresponding defeasible rule. Thus, we can represent this through the undercutting rule:

[*harry_british*—<*harry_born_in_bermuda*]⟵⊗[*harry_alien_parents*]

Definitions 5 and 6 impose a restriction on the rules that can appear in the head of backing and undercutting rules. In particular, the head of these rules can only be a defeasible rule and, therefore, it is not possible to attack or support strict rules. This restriction is necessary since, given that strict rules model indisputable information, they provide an unconditional connection between their antecedent (*Body*) and consequent (*Head*). The case regarding presumptions is different; since they are a particular case of defeasible rules, they can appear in the head of backing and undercutting rules.

Having defined all the elements that characterize the representational language of E-DeLP, we can now define the *extended defeasible logic programs* as follows.

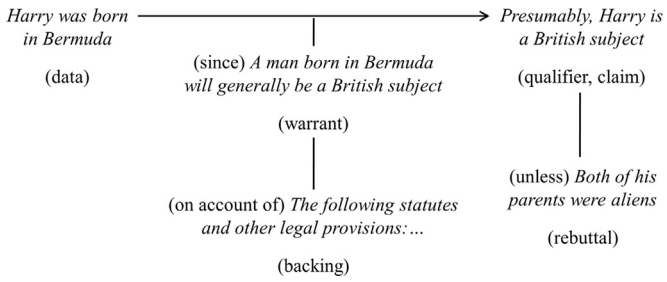

Fig. 1. Toulmin's "*Harry is a British subject*" famous example Toulmin (1958).

Definition 7 (*Extended defeasible logic program*). An *extended defeasible logic program* is a set $\mathcal{P}$ of facts, strict rules, defeasible rules, backing rules and undercutting rules. When convenient, an extended defeasible logic program $\mathcal{P}$ will be noted as $\mathcal{P} = (\Pi, \Delta, \Sigma)$, distinguishing the set Π of facts and strict rules, the set Δ of defeasible rules, and the set Σ of backing and undercutting rules.

It is important to notice that the existence of backing and undercutting rules for a defeasible rule in an E-DeLP program is not mandatory. In general, a defeasible rule without backing rules can be regarded as being always applicable, since there are not explicit requirements for its use. To illustrate this, let us consider the nature of Toulmin's warrants. According to Toulmin (1958) there may be warrants of different kinds, which provide different degrees of force on the connection between the data and claim of an argument. On the one hand, there may be warrants that authorize the acceptance of a claim unconditionally, in which case the connection expressed by the warrant is indisputable; given the characteristics of this kind of warrants, they are represented in E-DeLP using strict rules. On the other hand, there exist warrants that allow one to draw a conclusion tentatively, which are modeled in E-DeLP through defeasible rules. A warrant of the latter kind is illustrated in Toulmin's example about Harry (see Example 1), since it allows one to obtain the conclusion *presumably*. As mentioned in Toulmin (1958), for those warrants that could be challenged, backings may not always be explicitly specified. Thus, in such cases, there will be defeasible rules (corresponding to the warrants) without backing rules. Finally, the existence of undercutting rules for a defeasible rule R depends on the existence of conditions of exception for the warrant expressed by R.

The following example illustrates the extended defeasible logic program corresponding to the situation portrayed in Example 1.

Example 2. Let us consider the scenario introduced in Example 1. The extended defeasible logic program $\mathcal{P}_2 = (\Pi_2, \Delta_2, \Sigma_2)$ formalizes that scenario in E-DeLP, where

$$\Pi_2 = \{harry\_born\_in\_bermuda,\ british\_parliament\_acts,\ harry\_alien\_parents\}$$

$$\Delta_2 = \{harry\_british \text{—<} harry\_born\_in\_bermuda\}$$

$$\Sigma_2 = \left\{ \begin{array}{l} [harry\_british \text{—<} harry\_born\_in\_bermuda] \longleftarrow \oplus [british\_parliament\_acts] \\ [harry\_british \text{—<} harry\_born\_in\_bermuda] \longleftarrow \otimes [harry\_alien\_parents] \end{array} \right\}$$

The data "*harry_born_in_bermuda*", the backing "*british_parliament_acts*" and the rebuttal "*harry_alien_parents*" are represented as facts. The warrant is represented by the defeasible rule from Example 1. Finally, the backing and the rebuttal respectively provide reasons for and against the warrant, expressed through the backing and undercutting rules from Example 1.

Program rules in E-DeLP are ground, that is, they contain ground literals. However, in E-DeLP we will allow for the use of *schematic rules* with variables, which will be denoted using initial uppercase letters. Following the usual convention (Lifschitz, 1996), schematic rules will be instantiated by defining the set of all their ground instances. Thus, given a schematic rule R, *Ground*(R) will be the set of all ground instances of R. Similarly, given an E-DeLP program $\mathcal{P}$, *Ground*($\mathcal{P}$) will be the union of all sets *Ground* (R), where R is a rule in $\mathcal{P}$. Therefore, when using the knowledge expressed by an E-DeLP program $\mathcal{P}$, we will be implicitly considering the set *Ground*($\mathcal{P}$).

The use of schematic rules in E-DeLP allows for a better and more concise representation of different scenarios that could be modeled without them. To illustrate this, let us consider the characterization of warrants given by Toulmin, stating that they are

> "*general, hypothetical statements, which can act as bridges, and authorize the sort of step to which our particular argument commits us*" Toulmin (1958, p. 98)

Therefore, for instance, given Toulmin's example about Harry, the warrant expressing that "*a man born in Bermuda will generally be a British subject*" should be applicable not only for *Harry*, but for *any man* born in Bermuda. Hence, the general nature of Toulmin's warrants can be captured in E-DeLP by using schematic rules. To illustrate this, let us consider the following example.

Example 3. Let $\mathcal{P}_3 = (\Pi_3, \Delta_3, \Sigma_3)$ be the following extended defeasible logic program, which models the situation depicted in Example 1 using schematic rules.

$$\Pi_3 = \{born\_in\_bermuda(harry),\ british\_parliament\_acts,\ alien\_parents(harry)\}$$

$$\Delta_3 = \{british(X) \prec born\_in\_bermuda(X)\}$$

$$\Sigma_3 = \left\{ \begin{array}{l} [british(X) \prec born\_in\_bermuda(X)] \leftarrow \oplus [british\_parliament\_acts] \\ [british(X) \prec born\_in\_bermuda(X)] \leftarrow \otimes [alien\_parents(X)] \end{array} \right\}$$

In this case, the schematic defeasible rule captures the general nature of the warrant in Toulmin's example, being an inference rule that applies for all men born in Bermuda. Similarly, the schematic backing and undercutting rules provide reasons for or against the use of the corresponding defeasible rules.

Next, we introduce another E-DeLP program, modeling a scenario that will be used as a running example throughout the rest of the paper.

Example 4. Let $\mathcal{P}_4 = (\Pi_4, \Delta_4, \Sigma_4)$ be the following E-DeLP program:

$$\Pi_4 = \{night,\ electricity,\ lamp\_in\_room(l, r),\ switch\_on(l),\ broken\_lamp(l)\}$$

$$\Delta_4 = \left\{ \begin{array}{l} light\_on(X) \prec switch\_on(X) \\ illuminated\_room(X) \prec day \\ \sim illuminated\_room(X) \prec night \\ illuminated\_room(X) \prec night,\ lamp\_in\_room(Y, X),\ light\_on(Y) \end{array} \right\}$$

$$\Sigma_4 = \left\{ \begin{array}{l} [light\_on(X) \prec switch\_on(X)] \leftarrow \oplus [electricity] \\ [light\_on(X) \prec switch\_on(X)] \leftarrow \otimes [\sim electricity] \\ [light\_on(X) \prec switch\_on(X)] \leftarrow \oplus [\sim electricity,\ emergency\_lamp(X)] \\ [light\_on(X) \prec switch\_on(X)] \leftarrow \otimes [electricity,\ broken\_lamp(X)] \end{array} \right\}$$

Program $\mathcal{P}_4$ contains information with the aim of determining whether a room is illuminated or not. In particular, the first defeasible rule expresses that if the switch of a lamp is on, then there are reasons to believe that the light of the lamp is on. The remaining defeasible rules express alternative reasons to believe that a room will be illuminated (respectively, not illuminated). Given this scenario, the backing rules for the defeasible rule "$light\_on(X) \prec switch\_on(X)$" provide alternative conditions under which it is applicable. Analogously, the undercutting rules for the defeasible rule "$light\_on(X) \prec switch\_on(X)$" express alternative conditions against its use.

3. Definition of arguments

In this section we will introduce the elements required for defining a notion of *argument* in E-DeLP. In particular, given an extended defeasible logic program, it will be possible to obtain different types of arguments: *claim arguments*, *backing arguments*, and *undercutting arguments*. To achieve this, first we need to characterize a notion of derivation that allows for the construction of the aforementioned kinds of arguments.

Definition 8 (*Defeasible derivation*). Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program and L a ground literal. A *defeasible derivation* of L from $\mathcal{P}$ is a finite sequence of ground literals $L_1, L_2, \ldots, L_n = L$ such that every literal L_i $(1 \leq i \leq n)$ is in the sequence because

(a) L_i is a fact in Π;
(b) there exists a strict rule in Π with head L_i and body $B_1, \ldots, B_k$, where every literal B_j $(1 \leq j \leq k)$ is an element of the sequence such that $j < i$; or
(c) there exists a defeasible rule R in Δ with head L_i and body $B_1, \ldots, B_k$, where every literal B_j $(0 \leq j \leq k)$ is an element of the sequence such that $j < i$, and one of the following conditions holds:
  i. there is no backing rule in Σ with head R, or
  ii. there is a backing rule in Σ with head R and body $S_1, \ldots, S_m$, where every literal S_p $(1 \leq p \leq m)$ is an element of the sequence such that $p < i$.

As the preceding definition shows, the notion of derivation in E-DeLP only considers strict rules, defeasible rules, and backing rules. In particular, the reason why backing rules are accounted for in the derivation process relies on their nature. As mentioned before, according to Toulmin's model, backings establish the reasons why their associated warrants hold. Thus, in E-DeLP, a backing rule for a defeasible rule R determines conditions under which the warrant modeled by R is applicable. Moreover, if such conditions exist, they must be taken into account when using the defeasible rule R in the derivation process. As a result, condition (c) in Definition 8 states that if there are explicit requirements for using a defeasible rule R in a derivation, they must be satisfied.

Given an E-DeLP program, it can be the case that there exists more than one backing rule for a defeasible rule R. However, each backing rule for R expresses *alternative* conditions under which the warrant modeled by R is applicable. Thus, as stated by condition (c)ii in Definition 8, it suffices to have a derivation for the literals in the body of *one* backing rule for R. That is, it suffices to derive the literals establishing one of the alternative conditions under which the warrant modeled by R holds. Furthermore, since the existence of backing rules is not mandatory, it can be the case that a defeasible rule R has no backing rules. In such a case, captured by condition (c)i in Definition 8, it means that there are no explicit requirements for using the defeasible rule R and thus, it suffices to derive the literals in R's body. To illustrate this, let us consider the following example.

Example 5. Given the extended defeasible logic program $\mathcal{P}_4$ from Example 4, it is possible to obtain this derivation for the literal "illuminated_room(r)" : night, lamp_in_room(l,r), electricity, switch_on(l), light_on(l), illuminated_room(r). This derivation includes the literal "*electricity*" corresponding to the body of the first backing rule in $\mathcal{P}_4$. In particular, from $\mathcal{P}_4$ it is also possible to obtain a derivation for the literal "$\sim illuminated\_room(r)$", corresponding to the sequence: $night, \sim illuminated\_room(r)$.

According to Definition 8, extended defeasible derivations are obtained using facts, strict rules, defeasible rules and/or backing rules from an E-DeLP program. In particular, there may exist derivations that use only facts and/or strict rules. In those cases, we will say that they are *strict derivations*. For instance, given the program $\mathcal{P}_4$ from Example 4, every literal in the set Π_4 has a strict derivation composed of the literal itself.

As it can be observed in Example 5, from an E-DeLP program it is possible to obtain derivations for complementary literals[1] (with

[1] The complement of a literal $L = A$ with respect to the strong negation "$\sim$" is $\overline{L} = \sim A$. Similarly, the complement of a literal $L = \sim A$ with respect to "$\sim$" is $\overline{L} = A$.

respect to "$\sim$") which, clearly, are conflicting. Then, the conflict between complementary literals can be generalized when considering the set of facts and strict rules of an E-DeLP program. The following definition characterizes the notion of a *contradictory set*, identifying conflicts between literals.

Definition 9 (*Contradictory set*). Given an extended defeasible logic program $\mathcal{P}$, we will say that $C \subseteq \mathcal{P}$ is a *contradictory set* iff it is possible to obtain derivations for a literal L and its complement $\overline{L}$ from C.

For instance, if we consider the E-DeLP program $\mathcal{P}_4$ from Example 4, the set Π_4 is not contradictory. On the other hand, the set $\Pi_4 \cup \Delta_4 \cup \Sigma_4$ is contradictory since, as shown in Example 5, it allows one to obtain derivations for the complementary literals "*illuminated_room*(r)" and "$\sim$ *illuminated_room*(r)".

As mentioned at the beginning of this section, the notion of derivation provides a mechanism for the construction of arguments in E-DeLP. Definition 8 shows that, when obtaining a derivation for a literal, different kinds of rules may be used: strict rules, defeasible rules and backing rules. In contrast, undercutting rules are not used for obtaining defeasible derivations. Notwithstanding this, as will be shown next, undercutting rules will be used to build arguments against using defeasible rules. Then, given an E-DeLP program $\mathcal{P}$, it will be possible to build three different kinds of arguments. On the one hand, *claim arguments* for literals may be obtained. On the other hand, it will also be possible to build *backing arguments* and *undercutting arguments* which, respectively, express reasons for and against the use of defeasible rules.

Definition 10 (*Claim argument*). Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program and L a literal. A *claim argument* for L is a pair $\langle \mathcal{A}, L \rangle$, such that it satisfies the following conditions:

1. $\mathcal{A} \subseteq (\Delta \cup \Sigma)$,
2. there exists a derivation of L from $\Pi \cup \mathcal{A}$,
3. $\Pi \cup \mathcal{A}$ is a non-contradictory set, and
4. $\mathcal{A}$ is $\subset$−minimal: $\nexists \mathcal{B} \subset \mathcal{A}$ s.t. $\mathcal{B}$ satisfies conditions (2) and (3).

Intuitively, a claim argument $\mathcal{A}$ for a literal L is a minimal and non-contradictory set of defeasible rules and backing rules allowing one to obtain a derivation for L. Note that, although the first condition in Definition 10 would allow for the inclusion of undercutting rules in $\mathcal{A}$, these are excluded by condition (4) that establishes the minimality of the argument. This is because, as mentioned before, undercutting rules are not used in the derivation process. To illustrate the notion of claim argument, let us consider the following example.

Example 6. Given the extended defeasible logic program $\mathcal{P}_4$ from Example 4, it is possible to build the following claim arguments:

- $\langle \mathcal{A}_1, \textit{illuminated\_room}(r) \rangle$, with

$$\mathcal{A}_1 = \left\{ \begin{array}{l} \textit{illuminated\_room}(r) \text{—<} \textit{night}, \textit{lamp\_in\_room}(l, r), \textit{light\_on}(l) \\ \textit{light\_on}(l) \text{—<} \textit{switch\_on}(l) \\ [\textit{light\_on}(l) \text{—<} \textit{switch\_on}(l)] \text{←⊕} [\textit{electricity}] \end{array} \right\}$$

- $\langle \mathcal{A}_2, \sim \textit{illuminated\_room}(r) \rangle$, with $\mathcal{A}_2 = \{ \sim \textit{illuminated\_room}(r) \text{—<} \textit{night} \}$
- $\langle \mathcal{A}_3, \textit{light\_on}(l) \rangle$, with

$$\mathcal{A}_3 = \left\{ \begin{array}{l} \textit{light\_on}(l) \text{—<} \textit{switch\_on}(l) \\ [\textit{light\_on}(l) \text{—<} \textit{switch\_on}(l)] \text{←⊕} [\textit{electricity}] \end{array} \right\}$$

Also, for each literal in the set Π_4 we can build a claim argument with an empty set of rules: $\langle \varnothing, \textit{night} \rangle, \langle \varnothing, \textit{electricity} \rangle, \langle \varnothing, \textit{lamp\_in\_room}(l, r) \rangle, \langle \varnothing, \textit{switch\_on}(l) \rangle$ and $\langle \varnothing, \textit{broken\_lamp}(l) \rangle$.

The following definition characterizes the notion of *top defeasible rule* which, as will be shown in Section 5.2, will be used when specifying the BUAF associated to an E-DeLP program. Briefly, the top defeasible rule of an argument is the defeasible rule enabling to derive its conclusion.

Definition 11 (*Top defeasible rule*). Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program, $\langle \mathcal{A}, H \rangle$ a claim argument built from $\mathcal{P}$, and R:"H—<*Body*" a defeasible rule in Δ. If $R \in \mathcal{A}$, then we will say that R is a *top defeasible rule* of $\langle \mathcal{A}, H \rangle$.

As established by the preceding definition, the notion of top defeasible rule allows one to identify those arguments whose conclusion is the head of a defeasible rule. Hence, arguments whose conclusion is a fact in an E-DeLP program or is the head of a strict rule will not have a top defeasible rule. Moreover, as shown by the following proposition, each claim argument built from an E-DeLP program cannot have more than one top defeasible rule.

Proposition 1. *Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program and $\langle \mathcal{A}, H \rangle$ a claim argument built from $\mathcal{P}$. If R is a top defeasible rule of $\langle \mathcal{A}, H \rangle$, then $\nexists R' \in \Delta$ such that $R' \neq R$ and R' is a top defeasible rule of $\langle \mathcal{A}, H \rangle$.*

Proof. See Appendix.

Backing and undercutting arguments constitute positions in favor or against the use of defeasible rules. Therefore, differently from claim arguments, they will refer to the defeasible rule for which they respectively provide reasons for or against. In addition, to identify their nature, these two kinds of arguments will include the label '*b*' (backing) or '*u*' (undercutting).

Definition 12 (*Backing argument*). Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program and $R \in \Delta$ a defeasible rule. A *backing argument* for R is a pair $\langle \mathcal{A}, R \rangle_b$ such that $\mathcal{A} = \{[R] \text{←⊕} [L_1, \ldots, L_n]\} \cup \mathcal{A}'$ and it satisfies the following conditions:

1. $\mathcal{A} \subseteq (\Delta \cup \Sigma)$,
2. for every L_i $(1 \leq i \leq n)$ there exists a derivation from $\Pi \cup \mathcal{A}'$,
3. $\Pi \cup \mathcal{A}'$ is a non-contradictory set, and
4. $\mathcal{A}$ is $\subset$−minimal: $\nexists \mathcal{B} \subset \mathcal{A}'$ s.t. $\mathcal{B}$ satisfies conditions (2) and (3).

Definition 13 (*Undercutting argument*). Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program and $R \in \Delta$ a defeasible rule. An *undercutting argument* for R is a pair $\langle \mathcal{A}, R \rangle_u$ such that $\mathcal{A} = \{[R] \text{←⊗} [L_1, \ldots, L_n]\} \cup \mathcal{A}'$ and it satisfies the following conditions:

1. $\mathcal{A} \subseteq (\Delta \cup \Sigma)$,
2. for every L_i $(1 \leq i \leq n)$ there exists a derivation from $\Pi \cup \mathcal{A}'$,
3. $\Pi \cup \mathcal{A}'$ is a non-contradictory set, and
4. $\mathcal{A}$ is $\subset$−minimal: $\nexists \mathcal{B} \subset \mathcal{A}'$ s.t. $\mathcal{B}$ satisfies conditions (2) and (3).

Note that Definitions 12 and 13 are analogous. This is because the difference between these two forms of arguments relies on their nature of supporting or attacking a defeasible rule R, which is determined by the corresponding backing or undercutting rule of the argument. In particular, the first condition in Definitions 12 and 13 states that the backing rule or the undercutting rule that originates the argument, as well as any other defeasible or backing rule used for building it, must be a part of the program specification. Then, the second condition establishes that for building the argument it is necessary to obtain a derivation for every literal in the body of the corresponding backing or undercutting rule.

Finally, as stated by the third and fourth conditions, and similar to claim arguments, backing and undercutting arguments must be non-contradictory and minimal. To illustrate these notions, let us consider the following example.

Example 7. Given the extended defeasible logic program $\mathcal{P}_4$ from Example 4, it is possible to build the following backing and undercutting arguments:

- $\langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l) \rangle_b$, with

  $\mathcal{A}_4 = \{[light\_on(l) \prec switch\_on(l)] \leftarrow \oplus[electricity]\}$

- $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l) \rangle_u$, with

  $\mathcal{A}_5 = \{[light\_on(l) \prec switch\_on(l)] \leftarrow \otimes[electricity, broken\_lamp(l)]\}$

It is worth noticing that the three kinds of arguments characterized by Definitions 10, 12 and 13 are mutually exclusive. However, when convenient, we will abstract from the nature of an argument, referring to it simply as an argument. Taking this into account, we can define a notion of *sub-argument* that applies for all three kinds of arguments in E-DeLP.

Definition 14 (*Sub-argument*). Let $\mathcal{P}$ be an extended defeasible logic program and let $\langle \mathcal{A}, H \rangle$ and $\langle \mathcal{B}, Q \rangle$ be two arguments built from $\mathcal{P}$. We say that $\langle \mathcal{B}, Q \rangle$ is a *sub-argument* of $\langle \mathcal{A}, H \rangle$ (respectively, $\langle \mathcal{A}, H \rangle$ is a *super-argument* of $\langle \mathcal{B}, Q \rangle$) iff $\mathcal{B} \subseteq \mathcal{A}$. In particular, if $\mathcal{B}$ is a proper subset of $\mathcal{A}$ (*i.e.,* $\mathcal{B} \subset \mathcal{A}$), we say that $\langle \mathcal{B}, Q \rangle$ is a *proper sub-argument* of $\langle \mathcal{A}, H \rangle$.

The preceding definition establishes that every argument is a sub-argument of itself. However, the notion of proper sub-argument relates different arguments and, as will be shown in Section 4, will be useful to identify the conflicts between the arguments built from an E-DeLP program.

To illustrate the notion of sub-argument, let us consider Examples 6 and 7. All arguments from Example 6 that have an empty set of rules (*i.e.,* arguments whose conclusions are facts in $\mathcal{P}_4$) are sub-arguments of every argument built from $\mathcal{P}_4$. Also, $\langle \mathcal{A}_3, light\_on(l) \rangle$ is a sub-argument of $\langle \mathcal{A}_1, illuminated\_room(r) \rangle$. Moreover, the backing argument $\langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l) \rangle_b$ is a sub-argument of $\langle \mathcal{A}_3, light\_on(l) \rangle$ and $\langle \mathcal{A}_1, illuminated\_room(r) \rangle$ since they both make use of the defeasible rule "*light_on(l)–≺switch_on(l)*". The latter is formalized by the following proposition, which shows the relation between the notion of sub-argument and the existence of backing arguments in E-DeLP.

Proposition 2. *Let $\mathcal{P}$ be an extended defeasible logic program and $\langle \mathcal{A}, H \rangle$ an argument built from $\mathcal{P}$. If $\exists BR \in \mathcal{A}$ such that BR is a backing rule for a defeasible rule $R \in \mathcal{A}$, then there exists a backing argument $\langle \mathcal{B}, R \rangle_b$ such that $BR \in \mathcal{B}$ and $\langle \mathcal{B}, R \rangle_b$ is a sub-argument of $\langle \mathcal{A}, H \rangle$.*

Proof. See Appendix.

Given the characteristics of claim arguments, backing arguments, and undercutting arguments, by Definitions 10, 12 and 13 we know that they could contain at most one undercutting rule; more specifically, since the notion of defeasible derivation does not make use of undercutting rules, claim arguments and backing arguments will not contain any of those. On the other hand, an undercutting argument $\langle \mathcal{A}, R \rangle_u$ contains exactly one: the undercutting rule expressing reasons against the defeasible rule R. Then, it is possible to establish the following property on the sub-argument relation of E-DeLP.

Proposition 3. *Let $\mathcal{P}$ be an extended defeasible logic program and $\langle \mathcal{A}, H \rangle$ an argument built from $\mathcal{P}$. Every proper sub-argument of $\langle \mathcal{A}, H \rangle$ is either a claim argument or a backing argument.*

Proof. See Appendix.

4. Attacks between arguments

In this section we will present the different forms of conflict between arguments that can occur in E-DeLP, which lead to different forms of attack. As stated before, the use of strong negation "$\sim$" allows for the representation of complementary literals in E-DeLP. Moreover, given an E-DeLP program, it is possible to obtain derivations for complementary literals and build claim arguments for them; clearly, such arguments will be in conflict with each other since their conclusions are contradictory. Additionally, conflicts between arguments can appear when considering their conclusions together with the strict knowledge of an E-DeLP program and the resulting set is contradictory. Also, given the existence of undercutting arguments, we will identify a conflict between an undercutting argument for a defeasible rule R and any other argument that contains the defeasible rule R. Following these intuitions, and in accordance with other works on structured argumentation such as Prakken (2009), we will characterize the different forms of attack between arguments in E-DeLP, distinguishing between *rebutting attack*, *undercutting attack* and *undermining attack*. Specifically, rebutting attacks are directed at an intermediate or final conclusion of an argument; undercutting attacks dispute a defeasible rule of an argument; and undermining attacks are aimed at challenging a premise of an argument.

The following definition introduces the notion of *disagreement* between literals, which generalizes the conflict that arises from the consideration of complementary literals.

Definition 15 (*Disagreement*). Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program. We say that two literals L_1 and L_2 *disagree* iff the set $\Pi \cup \{L_1, L_2\}$ is contradictory.

The preceding definition establishes that there will be a conflict between every pair of literals such that when considered together with the set of strict knowledge of an E-DeLP program a contradictory set is obtained. Then, using this notion of disagreement, we can introduce *rebutting attack* between arguments in E-DeLP as follows.

Definition 16 (*Rebutting attack*). Let $\mathcal{P}$ be an extended defeasible logic program and let $\langle \mathcal{A}_1, H_1 \rangle$ and $\langle \mathcal{A}_2, H_2 \rangle$ be two arguments built from $\mathcal{P}$ such that $\langle \mathcal{A}_1, H_1 \rangle$ is a claim argument. We say that there exists a *rebutting attack* from $\langle \mathcal{A}_1, H_1 \rangle$ to $\langle \mathcal{A}_2, H_2 \rangle$ if there exists a claim sub-argument $\langle \mathcal{A}, H \rangle$ of $\langle \mathcal{A}_2, H_2 \rangle$ such that the literals H_1 and H disagree.

Given a rebutting attack from $\langle \mathcal{A}_1, H_1 \rangle$ to $\langle \mathcal{A}_2, H_2 \rangle$ we will also say that $\langle \mathcal{A}_1, H_1 \rangle$ *rebuts* $\langle \mathcal{A}_2, H_2 \rangle$. If there is no proper sub-argument $\langle \mathcal{B}, Q \rangle$ of $\langle \mathcal{A}, H \rangle$ such that the literals H_1 and Q disagree, we will say that $\langle \mathcal{A}, H \rangle$ is a *disagreement sub-argument* in this attack.

In particular, as will be shown in Section 5.2, the notion of disagreement sub-argument in an attack will be useful when characterizing the BUAF associated with an E-DeLP program. The following example illustrates the existence of rebutting attacks in E-DeLP, distinguishing the disagreement sub-argument in each case.

Example 8. Let us consider the extended defeasible logic program $\mathcal{P}_4$ from Example 4 and the claim arguments introduced in Example 6; in this case, the literals "*illuminated_room(r)*" and "$\sim$ *illuminated_room(r)*" disagree. Therefore, $\langle \mathcal{A}_1, illuminated\_room(r) \rangle$

rebuts $\langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle$ and vice-versa. In both cases, the disagreement sub-argument is, in particular, the argument being attacked.

As mentioned before, undercutting rules in an E-DeLP program are used for building undercutting arguments that constitute a position against the use of defeasible rules. Then, it is possible to identify a conflict between an undercutting argument for a defeasible rule R and those arguments that make use of R. In this way, the undercutting argument for R will originate an *undercutting attack* on those arguments that contain the defeasible rule R.

Definition 17 (*Undercutting attack*). Let $\mathcal{P}$ be an extended defeasible logic program and let $\langle \mathcal{A}, R\rangle$ and $\langle \mathcal{B}, H\rangle$ be two arguments built from $\mathcal{P}$ such that $\langle \mathcal{A}, R\rangle$ is an undercutting argument for the defeasible rule R. We say that there exists an *undercutting attack* from $\langle \mathcal{A}, R\rangle$ to $\langle \mathcal{B}, H\rangle$ if $R \in \mathcal{B}$.

Given an undercutting attack from $\langle \mathcal{A}, R\rangle$ to $\langle \mathcal{B}, H\rangle$, we will also say that $\langle \mathcal{A}, R\rangle$ *undercuts* $\langle \mathcal{B}, H\rangle$. If $\langle \mathcal{C}, Q\rangle$ is a sub-argument of $\langle \mathcal{B}, H\rangle$ with top defeasible rule R, then we will say that $\langle \mathcal{C}, Q\rangle$ is the *disagreement sub-argument* in this attack.

Definition 17 accounts for the conflict between the attacking argument $\langle \mathcal{A}, R\rangle$ and all sub-arguments of the attacked argument $\langle \mathcal{B}, H\rangle$ that contain the defeasible rule R. As a result, $\langle \mathcal{A}, R\rangle$ will undercut every sub-argument of $\langle \mathcal{B}, H\rangle$ that includes R. In addition, similar to rebutting attacks, the identification of the disagreement sub-argument in this kind of attack will be useful when characterizing the BUAF associated with an E-DeLP program in Section 5.2.

On the other hand, it can be noted that Definition 17 does not account for the existence of backing rules or backing arguments for the defeasible rule R. Therefore, undercutting attacks may exist even though the E-DeLP program has no backing rules for R. Notwithstanding this, as will be shown in Section 5, the existence of backing arguments will be taken into consideration in the resolution of this kind of attacks. To illustrate the notion of undercutting attack, let us consider the following example.

Example 9. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4. Let us consider the arguments built from $\mathcal{P}_4$, illustrated in Examples 6 and 7. Here, $\langle \mathcal{A}_5, light\_on(l) \multimap switch\_on(l)\rangle_u$ undercuts $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$. In addition, $\langle \mathcal{A}_5, light\_on(l) \multimap switch\_on(l)\rangle_u$ undercuts the sub-argument $\langle \mathcal{A}_3, light\_on(l)\rangle$ of $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$. In particular, in both attacks, the disagreement sub-argument is $\langle \mathcal{A}_3, light\_on(l)\rangle$.

Recall that arguments are built on the basis of premises. In E-DeLP, these premises are the facts and presumptions of an extended defeasible logic program. However, by condition (3) in Definitions 10, 12 and 13, arguments must be consistent with the strict knowledge of an E-DeLP program. Therefore, the only attackable premises of arguments in E-DeLP will be the presumptions. Hence, when attacking a presumption of an argument, we will say that an *undermining attack* occurs.

Definition 18 (*Undermining attack*). Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$ be an extended defeasible logic program and let $\langle \mathcal{A}_1, H_1\rangle$ and $\langle \mathcal{A}_2, H_2\rangle$ be two arguments built from $\mathcal{P}$ such that $\langle \mathcal{A}_1, H_1\rangle$ is a claim argument. We say that there exists an *undermining attack* from $\langle \mathcal{A}_1, H_1\rangle$ to $\langle \mathcal{A}_2, H_2\rangle$ if there exists a claim sub-argument $\langle \mathcal{A}, H\rangle$ of $\langle \mathcal{A}_2, H_2\rangle$ such that the literals H_1 and H disagree, and H is a presumption in $\mathcal{P}$, *i.e.*, $H \multimap \in \Delta$. Given an undermining attack from $\langle \mathcal{A}_1, H_1\rangle$ to $\langle \mathcal{A}_2, H_2\rangle$, we will also say that $\langle \mathcal{A}_1, H_1\rangle$ *undermines* $\langle \mathcal{A}_2, H_2\rangle$. If there is no proper sub-argument $\langle \mathcal{B}, Q\rangle$ of $\langle \mathcal{A}, H\rangle$ such that the literals H_1 and Q disagree, we will say that $\langle \mathcal{A}, H\rangle$ is a *disagreement sub-argument* in this attack.

From Definitions 16 and 18, it could be noticed that undermining attacks are considered as particular cases of rebutting attacks in E-DeLP. Recall that rebutting attacks in E-DeLP account for the disagreement between the conclusion of the attacking argument and the conclusion of a sub-argument from the attacked one. Thus, it can be the case that the conclusion of the disagreement sub-argument is a presumption, leading to an undermining attack. However, it should be noted that, although the characterization of these attacks in E-DeLP is similar, they still correspond to different forms of attack, as proposed in the literature.

Next, we will introduce a visual representation for arguments and attacks in E-DeLP. In this representation, arguments will be depicted using triangles, and triangles inside of bigger triangles will denote proper sub-arguments. In the case of claim arguments, the information located in the vertex of the triangle will denote the conclusion of the argument; for backing and undercutting arguments, the information located in the vertex of the triangle will denote the defeasible rule that is being supported or attacked. Also, for any argument, the information located within the triangle will correspond to the literals used for building the argument. The symbol '$\multimap$' will denote the connection between the literals used for building the argument through the use of defeasible rules. Similarly, the symbols '$\leftarrow\!\!\oplus$' and '$\leftarrow\!\!\otimes$' will denote the backing and undercutting rules used for building their corresponding arguments. In particular, claim arguments whose conclusion is obtained through strict derivations will be denoted by simply specifying their conclusion. Finally, attacks between arguments will be denoted using dashed arrows. To illustrate this visual representation, let us consider the following example.

Example 10. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4. Fig. 2 illustrates the arguments built from $\mathcal{P}_4$, corresponding to Examples 6 and 7. Also, the dashed arrows in Fig. 2 correspond to the attacks identified in Examples 8 and 9. On the one hand, as shown in Example 8, argument $\langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle$ rebuts argument $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$ and vice-

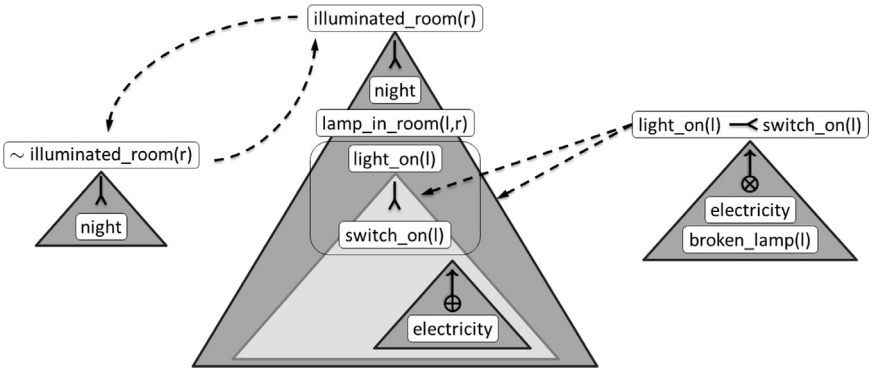

Fig. 2. Visual representation of arguments and attacks, corresponding to Example 10.

versa. On the other hand, as shown in Example 9, argument $\langle \mathcal{A}_5, light\_on(l) \multimap switch\_on(l) \rangle_u$ undercuts both $\langle \mathcal{A}_3, light\_on(l) \rangle$ and its super-argument $\langle \mathcal{A}_1, illuminated\_room(r) \rangle$.

5. Resolving attacks: defeats between arguments

The notion of attack introduced in Section 4 captures the different forms of conflict between arguments in E-DeLP. In the case of rebutting and undermining attacks, these conflicts arise from the use of strong negation "$\sim$", while for undercutting attacks, the conflicts originate in the existence of undercutting rules that provide reasons against the use of defeasible rules; however, the conflicts captured by the notion of attack will not always succeed. Therefore, we need to resolve these attacks to obtain the relation of *defeat* between arguments.

To determine the success or failure of attacks we need to perform some kind of evaluation over the conflicting arguments. Several approaches achieve this by considering a comparison criterion or a preference relation between arguments, used to decide which one of the conflicting arguments prevails; for instance, see Amgoud and Cayrol (2002), García and Simari (2004), Prakken (2009), and Amgoud and Vesic (2011). Similar to García and Simari (2004), in E-DeLP we will use a modular comparison criterion between arguments. In that way, the user of the system will be able to select the criterion that fits better according to the domain being represented.

Given the set of arguments built from an E-DeLP program $\mathcal{P}$, the sub-argument relation, the attacks between these arguments, and the adopted comparison criterion, we will specify a *Backing-Undercutting Argumentation Framework* (BUAF) Cohen et al. (2012) associated with $\mathcal{P}$. Then, given the associated BUAF, we will determine the defeats between arguments of $\mathcal{P}$, which will be used to perform the acceptability calculus on the arguments of $\mathcal{P}$. In particular, as will be shown in Section 6, this will allow for the application of different acceptability semantics among those proposed in the literature (for instance, see Dung, 1995; Baroni et al., 2011a).

In the following we will introduce the necessary background notions related to the BUAF proposed in Cohen et al. (2012). Then, we will show how to translate an E-DeLP program $\mathcal{P}$ into its associated BUAF. Finally, we will identify the defeats between arguments of $\mathcal{P}$, which will correspond to defeats in the associated BUAF.

5.1. Backing-Undercutting Argumentation Framework (BUAF)

In Cohen et al., 2012, the authors present the *Backing-Undercutting Argumentation Framework* (BUAF), a formalism that extends the representational capabilities of Dung's abstract argumentation framework (Dung, 1995) by adding a specialized support relation, a categorization of attacks, and a preference relation between arguments.

The support relation of the BUAF allows for the representation of different kinds of support between arguments. On the one hand, it is possible to represent the support that backings provide for their corresponding warrants, according to Toulmin's model for the layout of arguments (Toulmin, 1958). In addition, the BUAF accounts for existence of sub-arguments, thus providing a way for expressing the support links between sub-arguments and their super-arguments. Analogously, the attack relation of the BUAF allows for the consideration of different kinds of attack between arguments, distinguishing between rebutting attacks, undercutting attacks and undermining attacks. Finally, the BUAF makes use of a preference relation, for which no restrictions are imposed, that is used to resolve the conflicts between arguments. Thus, when two arguments $\mathcal{A}$ and $\mathcal{B}$ are related by the preference relation, *i.e.*, $(\mathcal{A}, \mathcal{B}) \in \preceq$, it means that argument $\mathcal{B}$ is at least as preferred as argument $\mathcal{A}$, and it is noted as $\mathcal{A} \preceq \mathcal{B}$. Then, following the usual notation, $\mathcal{A} \prec \mathcal{B}$ means $\mathcal{A} \preceq \mathcal{B}$ and $\mathcal{B} \preceq \mathcal{A}$. The formal characterization of the BUAF is given in the following definition.

Definition 19 (*Backing-Undercutting Argumentation Framework*). A *Backing-Undercutting Argumentation Framework* (BUAF) is a tuple $\langle \mathbb{A}, \mathbb{R}, \mathbb{S}, \preceq \rangle$, where

- $\mathbb{A}$ is a set of arguments;
- $\mathbb{R} = (\mathbb{R}_b, \mathbb{U}_c, \mathbb{U}_m)$ represents an attack relation such that $\mathbb{R}_b \subseteq \mathbb{A} \times \mathbb{A}$ denotes rebutting attacks, $\mathbb{U}_c \subseteq \mathbb{A} \times \mathbb{A}$ denotes undercutting attacks, $\mathbb{U}_m \subseteq \mathbb{A} \times \mathbb{A}$ denotes undermining attacks, and the sets $\mathbb{R}_b$, $\mathbb{U}_c$, and $\mathbb{U}_m$ are pairwise disjoint;
- $\mathbb{S} = (\mathbb{B}_k, \mathbb{S}_{\sqsubseteq})$ represents a support relation such that $\mathbb{B}_k \subseteq \mathbb{A} \times \mathbb{A}$ denotes support provided by backings and $\mathbb{S}_{\sqsubseteq} \subseteq \mathbb{A} \times \mathbb{A}$ denotes support provided by sub-arguments;
- $\preceq \subseteq \mathbb{A} \times \mathbb{A}$ is a preference relation; and
- there are no arguments $\mathcal{A}, \mathcal{B} \in \mathbb{A}$ such $\mathcal{A}$ simultaneously attacks and supports $\mathcal{B}$ (*i.e.*, $(\mathbb{R}_b \cup \mathbb{U}_c \cup \mathbb{U}_m) \cap (\mathbb{B}_k \cup \mathbb{S}_{\sqsubseteq}) = \varnothing$).

Rebutting and undermining attacks in a BUAF can be resolved directly by applying preferences on the conflicting arguments. In this way, they will lead to their corresponding forms of defeat when the attacking arguments are not less preferred than the attacked ones; in contrast, undercutting attacks require the consideration of backing arguments (if they exist) for the attacked arguments. These defeats are identified in Cohen et al. (2012) as *primary defeats*, and are characterized by the following definition.

Definition 20 (*Primary defeat*). Let $\langle \mathbb{A}, \mathbb{R}, \mathbb{S}, \preceq \rangle$ be a BUAF and $\mathcal{A}, \mathcal{C} \in \mathbb{A}$. $\mathcal{A}$ *primarily defeats* $\mathcal{C}$ iff one of the following conditions holds:

- $(\mathcal{A}, \mathcal{C}) \in (\mathbb{R}_b \cup \mathbb{U}_m)$ and $\mathcal{A} \not\prec \mathcal{C}$,
- $(\mathcal{A}, \mathcal{C}) \in \mathbb{U}_c$ and $\nexists \mathcal{B} \in \mathbb{A}$ such that $(\mathcal{B}, \mathcal{C}) \in \mathbb{B}_k$, or
- $(\mathcal{A}, \mathcal{C}) \in \mathbb{U}_c$ and $\exists \mathcal{B} \in \mathbb{A}$ such that $(\mathcal{B}, \mathcal{C}) \in \mathbb{B}_k$ and $\mathcal{A} \not\prec \mathcal{B}$.

The preceding definition provides a mechanism for the resolution of the conflicts expressed in the attack relation of the BUAF. However, there might be other conflicts between arguments that are not explicitly specified in the attack relation of the BUAF. Several works in the literature of support in argumentation have discussed the need for combining attacks and supports in order to define new kinds of attacks (see *e.g.* Cayrol and Lagasquie-Schiex, 2005; Boella et al., 2010; Nouioua and Risch, 2011; Cayrol and Lagasquie-Schiex, 2013), which the authors of Cayrol and Lagasquie-Schiex (2013) refer to as *complex attacks*. Following this intuition, the opposing nature of backing and undercutting arguments in a BUAF suggests the existence of a conflict between them. This implicit conflict can be appreciated in Definition 20, since the resolution of undercutting attacks involves a comparison between the attacking argument and a backing argument for the attacked one. To model this kind of conflict explicitly, in Cohen et al. (2012) the authors define the notion of *implicit defeat* as follows.

Definition 21 (*Implicit defeat*). Given a BUAF $\langle \mathbb{A}, \mathbb{R}, \mathbb{S}, \preceq \rangle$ and arguments $\mathcal{A}, \mathcal{B}, \mathcal{C} \in \mathbb{A}$. If $(\mathcal{B}, \mathcal{C}) \in \mathbb{B}_k$ and $(\mathcal{A}, \mathcal{C}) \in \mathbb{U}_c$, then

- $\mathcal{A}$ *implicitly defeats* $\mathcal{B}$ iff $\mathcal{A} \not\prec \mathcal{B}$; and
- $\mathcal{B}$ *implicitly defeats* $\mathcal{A}$ iff $\mathcal{B} \not\prec \mathcal{A}$.

Definition 21 addresses the implicit conflict between backing and undercutting arguments in a BUAF by determining the conditions under which they defeat one another; however, given the nature

of backing arguments, it is necessary to account for additional conflicts that may arise. That is, backing arguments establish the conditions under which their associated warrants hold. Therefore, if a backing $\mathcal{B}$ of an argument $\mathcal{A}$ is defeated, then the conditions that justify $\mathcal{A}$'s warrant no longer hold and thus, $\mathcal{A}$ should also be defeated. Similarly, since the support relation of the BUAF accounts for the notion of sub-argument, defeats on arguments should be propagated to their corresponding super-arguments. Following these intuitions, in Cohen et al. (2012) the authors characterize the notion of *indirect defeat*, which propagates defeats on arguments to the arguments they support.

Definition 22 (*Indirect defeat*). Let $\langle \mathbb{A}, \mathbb{R}, \mathbb{S}, \preceq \rangle$ be a BUAF and $\mathcal{A}, \mathcal{C} \in \mathbb{A}$. $\mathcal{A}$ *indirectly defeats* $\mathcal{C}$ iff $\exists \mathcal{B} \in \mathbb{A}$ such that $(\mathcal{B}, \mathcal{C}) \in (\mathbb{B}_k \cup \mathbb{S}_{\sqsubset})$ and $\mathcal{A}$ primarily defeats, implicitly defeats, or indirectly defeats $\mathcal{B}$.

As proposed in Cohen et al. (2012), the following definition groups the different kinds of defeat that can occur between the arguments of a BUAF, providing a unified notation for them.

Definition 23 (*Defeat*). Let $\langle \mathbb{A}, \mathbb{R}, \mathbb{S}, \preceq \rangle$ be a BUAF and $\mathcal{A}, \mathcal{B} \in \mathbb{A}$. $\mathcal{A}$ *defeats* $\mathcal{B}$, noted as $\mathcal{A} \longrightarrow \mathcal{B}$, iff $\mathcal{A}$ primarily defeats, implicitly defeats, or indirectly defeats $\mathcal{B}$.

Having formally defined the BUAF and the defeats between arguments, in Cohen et al. (2012) the authors show that it is possible to characterize a Dung-like abstract argumentation framework (AF) (Dung, 1995) by using the set of arguments of the BUAF and the defeats between them. We will refer to this AF as the AF *associated with the* BUAF. In addition, as shown in Cohen et al. (2012), the associated AF is such that it accepts exactly the same arguments as the BUAF under a given semantics. These intuitions are formalized in the following definition.

Definition 24 (AF *associated with a* BUAF, BUAF *extensions*). Let $\text{BUAF} = \langle \mathbb{A}, \mathbb{R}, \mathbb{S}, \preceq \rangle$ be a backing-undercutting argumentation framework. The *abstract argumentation framework associated with* BUAF is $\text{AF} = \langle \mathbb{A}, \longrightarrow \rangle$, where $\longrightarrow$ is the defeat relation given in Definition 23. Then, E is an extension of BUAF under a semantics $s \in \{complete, preferred, stable, grounded\}$ iff E is an extension of AF under s.

The following definitions introduce a series of semantic notions from Dung (1995) leading to the characterization of different extensions of an AF. In particular, following the approach adopted in Cohen et al. (2012), in this work we will only consider the *complete*, *preferred*, *stable* and *grounded* semantics from Dung (1995); however, it should be noted that these results can be extended to other acceptability semantics such as semi-stable (Caminada, 2006) or ideal (Dung et al., 2007). Each one of these semantics characterizes a set of extensions, identifying sets of arguments that verify certain properties and can be collectively accepted. In that way, an extension E of an AF under a semantics $s \in \{complete, preferred, stable, grounded\}$ will be a set of accepted arguments of the AF satisfying the constraints imposed by the corresponding semantics.

Definition 25 (*Conflict-freeness, acceptability and admissibility*). Let $\langle \mathbb{A}, \longrightarrow \rangle$ be an AF and $S \subseteq \mathbb{A}$:

- S is conflict-free iff $\nexists \mathcal{A}, \mathcal{B} \in S$ s.t. $\mathcal{A} \longrightarrow \mathcal{B}$.
- $\mathcal{A}$ is acceptable with respect to S iff $\forall \mathcal{B} \in \mathbb{A}$ s.t. $\mathcal{B} \longrightarrow \mathcal{A}$, $\exists \mathcal{C} \in S$ s.t. $\mathcal{C} \longrightarrow \mathcal{B}$.
- S is *admissible* iff it is conflict-free and $\forall \mathcal{A} \in S$ it holds that $\mathcal{A}$ is acceptable with respect to S.

Definition 26 (AF *extensions*). Let $\langle \mathbb{A}, \longrightarrow \rangle$ be an AF and $S \subseteq \mathbb{A}$:

- S is a *complete extension* of AF iff it is admissible and $\forall \mathcal{A} \in \mathbb{A}$: if $\mathcal{A}$ is acceptable with respect to S, then $\mathcal{A} \in S$.
- S is a *preferred extension* of AF iff it is a maximal (with respect to $\subseteq$) admissible set.
- S is a *stable extension* of AF iff it is conflict-free and $\forall \mathcal{A} \in \mathbb{A} \setminus S$: $\exists \mathcal{B} \in S$ s.t. $\mathcal{B} \longrightarrow \mathcal{A}$.
- S is the *grounded extension* of AF iff it is the smallest (with respect to $\subseteq$) complete extension.

5.2. BUAF *associated with an E-DeLP program*

We will now explain the intuitions behind the process of translating an E-DeLP program into its associated BUAF, followed by its formal definition. That is, we will determine how to obtain the four elements that provide the specification of the associated BUAF: the set of arguments $\mathbb{A}$, the attack relation $\mathbb{R} = (\mathbb{R}_b, \mathbb{U}_c, \mathbb{U}_m)$, the support relation $\mathbb{S} = (\mathbb{B}_k, \mathbb{S}_{\sqsubset})$ and the preference relation $\preceq$.

The first element of the BUAF associated with an E-DeLP program is the set of arguments $\mathbb{A}$. As presented in Section 3, Definitions 10, 12 and 13 provide a mechanism for obtaining all arguments that can be built from an extended defeasible logic program. Thus, the set of arguments $\mathbb{A}$ of the associated BUAF will coincide with the set of arguments built from its corresponding E-DeLP program.

The attack relation $\mathbb{R} = (\mathbb{R}_b, \mathbb{U}_c, \mathbb{U}_m)$ of the associated BUAF will be characterized by considering the different forms of attack introduced in Definitions 16–18; since the sets $\mathbb{R}_b$, $\mathbb{U}_c$, and $\mathbb{U}_m$ of $\mathbb{R}$ must be pairwise disjoint, each attack in E-DeLP will belong to only one of these sets in the associated BUAF. Rebutting and undercutting attacks in E-DeLP will be mapped into the corresponding sets $\mathbb{R}_b$ and $\mathbb{U}_c$ of the associated BUAF. Recall that undermining attacks in E-DeLP are also rebutting attacks; however, they cannot be mapped into both $\mathbb{R}_b$ and $\mathbb{U}_m$. Thus, as undermining attacks are a particular case of rebutting attacks, they will be mapped into the set $\mathbb{U}_m$ of the associated BUAF.

According to Definitions 16–18, given an attack (rebutting, undercutting or undermining) from an argument $\mathcal{A}$ to an argument $\mathcal{B}$ in E-DeLP, there will also exist a derived attack (of the same kind) from $\mathcal{A}$ to every super-argument $\mathcal{C}$ of $\mathcal{B}$. However, when specifying the attack relation of the associated BUAF we will not include the derived attacks that arise from the consideration of super-arguments. That is to say, given an attack from an argument $\mathcal{A}$ to an argument $\mathcal{B}$ in an E-DeLP program, the attack relation of the associated BUAF will only include an attack from $\mathcal{A}$ to $\mathcal{D}$, where $\mathcal{D}$ is a disagreement sub-argument in the attack from $\mathcal{A}$ to $\mathcal{B}$. Then, as will become clear in Section 5.3, the conflicts involving the super-arguments will be captured by the existence of indirect defeats (see Definition 22) in the associated BUAF.

Let us now consider the specification of the support relation $\mathbb{S} = (\mathbb{B}_k, \mathbb{S}_{\sqsubset})$ of the associated BUAF, which distinguishes between the support provided by backings (set $\mathbb{B}_k$) and the support provided by sub-arguments (set $\mathbb{S}_{\sqsubset}$). Given an E-DeLP program $\mathcal{P}$ and a backing argument $\mathcal{B}$ for a defeasible rule R, $\mathcal{B}$ provides support for every argument $\mathcal{A}$ built from $\mathcal{P}$ that contains the rule R. Notwithstanding this, similar to the attacks, the associated BUAF will only include the "direct" support links between arguments. That is, in the case of backings, the set $\mathbb{B}_k$ will only include the support links between $\mathcal{B}$ and those arguments $\mathcal{A}$ whose top defeasible rule is R. Then, the support that $\mathcal{B}$ provides for the super-arguments of $\mathcal{A}$ will be captured by the sub-argument relation between $\mathcal{A}$ and its super-arguments.

Regarding the specification of the set $\mathbb{S}_{\sqsubset}$, the associated BUAF will be such that it considers only the relevant sub-argument relations between arguments. Thus, it will be determined using the notion of proper sub-argument characterized in Definition 14.

Moreover, similar to the set $\mathbb{B}_k$, the sub-argument relation in the associated BUAF will only represent the "direct" links between arguments and their proper sub-arguments. That is, given an argument $\mathcal{A}$ and a proper sub-argument $\mathcal{B}$ of $\mathcal{A}$, the set $\mathbb{S}_{\sqsubset}$ in the associated BUAF will include a pair $(\mathcal{B},\mathcal{A})$ iff there is no proper sub-argument $\mathcal{C}$ of $\mathcal{A}$ such that $\mathcal{B}$ is a proper sub-argument of $\mathcal{C}$. Then, the support links between $\mathcal{B}$ and the super-arguments of $\mathcal{A}$ will be captured by chaining support links in the associated BUAF.

Since we are interested in including only the relevant sub-argument links within the set $\mathbb{S}_{\sqsubset}$, we will take additional considerations. On the one hand, given the characterization of arguments in E-DeLP, backing arguments will be, in particular, proper sub-arguments of the arguments they support. Then, given a pair $(\mathcal{B},\mathcal{A})$ in the set $\mathbb{B}_k$ of the associated BUAF, we will not include the pair $(\mathcal{B},\mathcal{A})$ in the set $\mathbb{S}_{\sqsubset}$. The reason for this is that we want the associated BUAF to include the most specific relations between arguments, because backings are a special case of sub-arguments in E-DeLP. On the other hand, as it was shown in Section 3, for every literal L that has a strict derivation from an E-DeLP program, there will exist a claim argument $\mathcal{A}$, where $\mathcal{A}$ is an empty set of rules. In that way, arguments like $\mathcal{A}$ will be proper sub-arguments of any other argument built from that program. Therefore, given such an argument $\mathcal{A}$ for L, since we want to avoid irrelevant links between arguments in the associated BUAF, the set $\mathbb{S}_{\sqsubset}$ will only include a pair $(\mathcal{A},\mathcal{B})$ if the literal L is used in the derivation process that leads to obtaining argument $\mathcal{B}$.

The preference relation $\preceq$ of the associated BUAF will be specified in terms of the comparison criterion adopted by the E-DeLP program, which might lead to incomparable arguments. Thus, the preference relation of the BUAF will inherit the characteristics of the comparison criterion used in E-DeLP.

Next, based on the intuitions presented above, we provide a formal characterization of the BUAF associated with an E-DeLP program $\mathcal{P}$. This BUAF will be the *backing-undercutting argumentation framework associated with* $\mathcal{P}$, and will be denoted as $\mathsf{BUAF}_{\mathcal{P}}$.

Definition 27 (BUAF *associated with an E-DeLP Program*). Let $\mathcal{P}$ be an extended defeasible logic program. The *backing-undercutting argumentation framework associated with* $\mathcal{P}$ is $\mathsf{BUAF}_{\mathcal{P}} = \langle \mathbb{A}_{\mathcal{P}}, \mathbb{R}_{\mathcal{P}}, \mathbb{S}_{\mathcal{P}}, \preceq_{\mathcal{P}} \rangle$, where

- $\mathbb{A}_{\mathcal{P}}$ is the set of arguments built from $\mathcal{P}$, according to Definitions 10, 12 and 13.
- $\mathbb{R}_{\mathcal{P}} = \mathbb{R}_{b\mathcal{P}} \cup \mathbb{U}_{c\mathcal{P}} \cup \mathbb{U}_{m\mathcal{P}}$ is obtained from Definitions 16–18 as follows:
  - Let $\langle \mathcal{A}_1, H_1\rangle, \langle \mathcal{A}_2, H_2\rangle \in \mathbb{A}_{\mathcal{P}}$ s.t. $\langle \mathcal{A}_1, H_1\rangle$ rebuts $\langle \mathcal{A}_2, H_2\rangle$ and the disagreement sub-argument in the attack is $\langle \mathcal{A}, H\rangle$. If $\langle \mathcal{A}_1, H_1\rangle$ also undermines $\langle \mathcal{A}_2, H_2\rangle$, then $(\langle \mathcal{A}_1, H_1\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{U}_{m\mathcal{P}}$. Otherwise, if $\langle \mathcal{A}_1, H_1\rangle$ does not undermine $\langle \mathcal{A}_2, H_2\rangle$, then $(\langle \mathcal{A}_1, H_1\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{R}_{b\mathcal{P}}$.
  - Let $\langle \mathcal{A}, R\rangle, \langle \mathcal{B}, H\rangle \in \mathbb{A}_{\mathcal{P}}$ s.t. $\langle \mathcal{A}, R\rangle$ undercuts $\langle \mathcal{B}, H\rangle$. If $\langle \mathcal{C}, Q\rangle$ is the disagreement sub-argument in this attack, then, $(\langle \mathcal{A}, R\rangle, \langle \mathcal{C}, Q\rangle) \in \mathbb{U}_{c\mathcal{P}}$.
- $\mathbb{S}_{\mathcal{P}} = \mathbb{B}_{k\mathcal{P}} \cup \mathbb{S}_{\sqsubset\mathcal{P}}$ is obtained from Definition 14 and Proposition 3 as follows[2]:
  - Let $\langle \mathcal{B}, R\rangle, \langle \mathcal{A}, H\rangle \in \mathbb{A}_{\mathcal{P}}$ s.t. $\langle \mathcal{B}, R\rangle$ is a backing argument for the defeasible rule R and is a proper sub-argument of $\langle \mathcal{A}, H\rangle$. If R is the top defeasible rule of $\langle \mathcal{A}, H\rangle$, then $(\langle \mathcal{B}, R\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{B}_{k\mathcal{P}}$.
  - Let $\langle \mathcal{A}, H\rangle, \langle \mathcal{B}, Q\rangle \in \mathbb{A}_{\mathcal{P}}$ s.t. $\langle \mathcal{A}, H\rangle$ is a proper sub-argument of $\langle \mathcal{B}, Q\rangle$. If $\langle \mathcal{A}, H\rangle$ is a claim argument s.t. the literal H is used in the derivation process for building $\langle \mathcal{B}, Q\rangle$ and there is no proper sub-argument $\langle \mathcal{C}, P\rangle$ of $\langle \mathcal{B}, Q\rangle$ s.t. $\langle \mathcal{A}, H\rangle$ is a proper sub-argument of $\langle \mathcal{C}, P\rangle$, then $(\langle \mathcal{A}, H\rangle, \langle \mathcal{B}, Q\rangle) \in \mathbb{S}_{\sqsubset\mathcal{P}}$.

[2] Recall that, by Proposition 3, every proper sub-argument in E-DeLP is either a claim argument or a backing argument.

- The preference relation $\preceq_{\mathcal{P}}$ exactly reflects the comparison criterion adopted in $\mathcal{P}$.

The following example illustrates the process for obtaining the BUAF associated with an E-DeLP program, as characterized by the previous definition.

Example 11. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4. The backing-undercutting argumentation framework associated with $\mathcal{P}_4$ is $\mathsf{BUAF}_{\mathcal{P}_4} = \langle \mathbb{A}_{\mathcal{P}_4}, \mathbb{R}_{\mathcal{P}_4}, \mathbb{S}_{\mathcal{P}_4}, \preceq_{\mathcal{P}_4} \rangle$, where

- $\mathbb{A}_{\mathcal{P}_4} = \left\{ \begin{array}{ll} \langle \mathcal{A}_1, illuminated\_room(r)\rangle & \langle \emptyset, switch\_on(l)\rangle \\ \langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle & \langle \emptyset, night\rangle \\ \langle \mathcal{A}_3, light\_on(l)\rangle & \langle \emptyset, lamp\_in\_room(l,r)\rangle \\ \langle \mathcal{A}_4, light\_on(l) \multimap switch\_on(l)\rangle_b & \langle \emptyset, electricity\rangle \\ \langle \mathcal{A}_5, light\_on(l) \multimap switch\_on(l)\rangle_u & \langle \emptyset, broken\_lamp(l)\rangle \end{array} \right\}$

$$\mathcal{A}_1 = \left\{ \begin{array}{l} illuminated\_room(r) \multimap night, lamp\_in\_room(l,r), light\_on(l) \\ light\_on(l) \multimap switch\_on(l) \\ [light\_on(l) \multimap switch\_on(l)] \leftarrow \oplus [electricity] \end{array} \right\}$$

$$\mathcal{A}_2 = \{ \sim illuminated\_room(r) \multimap night \}$$

$$\mathcal{A}_3 = \left\{ \begin{array}{l} light\_on(l) \multimap switch\_on(l) \\ [light\_on(l) \multimap switch\_on(l)] \leftarrow \oplus [electricity] \end{array} \right\}$$

$$\mathcal{A}_4 = \{ [light\_on(l) \multimap switch\_on(l)] \leftarrow \oplus [electricity] \}$$

$$\mathcal{A}_5 = \{ [light\_on(l) \multimap switch\_on(l)] \leftarrow \otimes [electricity, broken\_lamp(l)] \}$$

That is, $\mathbb{A}_{\mathcal{P}_4}$ includes the arguments identified in Examples 6 and 7.

- $\mathbb{R}_{\mathcal{P}_4} = \mathbb{R}_{b\mathcal{P}_4} \cup \mathbb{U}_{c\mathcal{P}_4} \cup \mathbb{U}_{m\mathcal{P}_4}$, with

$$\mathbb{R}_{b\mathcal{P}_4} = \left\{ \begin{array}{l} (\langle \mathcal{A}_1, illuminated\_room(r)\rangle, \langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle) \\ (\langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle, \langle \mathcal{A}_1, illuminated\_room(r)\rangle) \end{array} \right\}$$

$$\mathbb{U}_{c\mathcal{P}_4} = \{ (\langle \mathcal{A}_5, light\_on(l) \multimap switch\_on(l)\rangle_u, \langle \mathcal{A}_3, light\_on(l)\rangle) \}$$

$$\mathbb{U}_{m\mathcal{P}_4} = \emptyset$$

In particular, the set $\mathbb{R}_{b\mathcal{P}_4}$ includes the rebutting attacks from Example 8. Similarly, the set $\mathbb{U}_{c\mathcal{P}_4}$ includes the undercutting attack from Example 9.

- $\mathbb{S}_{\mathcal{P}_4} = \mathbb{B}_{k\mathcal{P}_4} \cup \mathbb{S}_{\sqsubset\mathcal{P}_4}$, with

$$\mathbb{B}_{k\mathcal{P}_4} = \{ (\langle \mathcal{A}_4, light\_on(l) \multimap switch\_on(l)\rangle_b, \langle \mathcal{A}_3, light\_on(l)\rangle) \}$$

$$\mathbb{S}_{\sqsubset\mathcal{P}_4} = \left\{ \begin{array}{l} (\langle \mathcal{A}_3, light\_on(l)\rangle, \langle \mathcal{A}_1, illuminated\_room(r)\rangle) \\ (\langle \emptyset, switch\_on(l)\rangle, \langle \mathcal{A}_4, light\_on(l) \multimap switch\_on(l)\rangle_b) \\ (\langle \emptyset, night\rangle, \langle \mathcal{A}_1, illuminated\_room(r)\rangle) \\ (\langle \emptyset, night\rangle, \langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle) \\ (\langle \emptyset, lamp\_in\_room(l,r)\rangle, \langle \mathcal{A}_1, illuminated\_room(r)\rangle) \\ (\langle \emptyset, electricity\rangle, \langle \mathcal{A}_4, light\_on(l) \multimap switch\_on(l)\rangle_b) \\ (\langle \emptyset, electricity\rangle, \langle \mathcal{A}_5, light\_on(l) \multimap switch\_on(l)\rangle_u) \\ (\langle \emptyset, broken\_lamp(l)\rangle, \langle \mathcal{A}_5, light\_on(l) \multimap switch\_on(l)\rangle_u) \end{array} \right\}$$

The set $\mathbb{B}_{k\mathcal{P}_4}$ models the support that the backing argument $\langle \mathcal{A}_4, light\_on(l) \multimap switch\_on(l)\rangle_b$ provides for the claim argument $\langle \mathcal{A}_3, light\_on(l)\rangle$, since the latter is the super-argument of the former whose top defeasible rule is "$light\_on(l) \multimap switch\_on(l)$". Then, the set $\mathbb{S}_{\sqsubset\mathcal{P}_4}$ models the proper sub-argument relation between $\langle \mathcal{A}_3, light\_on(l)\rangle$ and $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$, as well as those resulting from the consideration of the arguments whose conclusions are facts in $\mathcal{P}_4$.

- Let us suppose that the argument comparison criterion adopted by $\mathcal{P}_4$ is such that $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$ is preferred to $\langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle$ and, on the other hand, $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ is preferred to $\langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l)\rangle_b$. Also, suppose that the comparison criterion does not establish other preferences on the arguments built from $\mathcal{P}_4$. Then, the preference relation of $\mathsf{BUAF}_{\mathcal{P}_4}$ is

$$\preceq_{\mathcal{P}_4} = \left\{ \begin{array}{l} (\langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle, \langle \mathcal{A}_1, illuminated\_room(r)\rangle) \\ (\langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l)\rangle_b, \langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u) \end{array} \right\}$$

Fig. 3 illustrates the graphical representation of $\mathsf{BUAF}_{\mathcal{P}_4}$, as introduced in Cohen et al. (2012). In this representation, the attack relation of a BUAF is denoted using a tailed arrow '$\rightarrowtail$'. In particular, the tailed arrows are labeled with 'R_b', 'U_c' and 'U_m', depending on the attack they represent. Similarly, the support relation of a BUAF is denoted using '$\Rightarrow$', where the double arrows are respectively labeled with 'b' and '$\sqsubseteq$' to distinguish between the support provided by backings and sub-arguments. In addition, for legibility purposes, the arguments belonging to the set $\mathbb{A}_{\mathcal{P}_4}$ are depicted in Fig. 3 as $Arg_1, Arg_2, \ldots, Arg_n$, where

$Arg_1 = \langle \mathcal{A}_1, illuminated\_room(r)\rangle$, $Arg_6 = \langle \emptyset, switch\_on(l)\rangle$
$Arg_2 = \langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle$, $Arg_7 = \langle \emptyset, night\rangle$
$Arg_3 = \langle \mathcal{A}_3, light\_on(l)\rangle$, $Arg_8 = \langle \emptyset, lamp\_in\_room(l,r)\rangle$
$Arg_4 = \langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l)\rangle_b$, $Arg_9 = \langle \emptyset, electricity\rangle$
$Arg_5 = \langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$, $Arg_{10} = \langle \emptyset, broken\_lamp(l)\rangle$

As the following proposition shows, given the transformation characterized by Definition 27, the $\mathsf{BUAF}_{\mathcal{P}}$ associated with an E-DeLP program $\mathcal{P}$ is such that every argument in $\mathsf{BUAF}_{\mathcal{P}}$ has at most one backing argument.

Proposition 4. *Let $\mathcal{P}$ be an extended defeasible logic program and $\mathsf{BUAF}_{\mathcal{P}} = \langle \mathbb{A}_{\mathcal{P}}, \mathbb{R}_{\mathcal{P}}, \mathbb{S}_{\mathcal{P}}, \preceq_{\mathcal{P}}\rangle$ the backing-undercutting argumentation framework associated with $\mathcal{P}$. For every $\langle \mathcal{A}, H\rangle \in \mathbb{A}_{\mathcal{P}}$ it holds that if $\exists \langle \mathcal{B}, R\rangle \in \mathbb{A}_{\mathcal{P}}$ such that $(\langle \mathcal{B}, R\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{B}_{k\mathcal{P}}$, then $\nexists \langle \mathcal{C}, R'\rangle \in \mathbb{A}_{\mathcal{P}}$ such that $\langle \mathcal{C}, R'\rangle \neq \langle \mathcal{B}, R\rangle$ and $(\langle \mathcal{C}, R'\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{B}_{k\mathcal{P}}$.*

Proof. See Appendix.

5.3. Defeats in E-DeLP

As mentioned at the beginning of this section, defeats between arguments built from an E-DeLP program will correspond to defeats in the BUAF associated with that program, as shown by the following definition.

Definition 28 (*Defeat in E-DeLP*). Let $\mathcal{P}$ be an extended defeasible logic program and $\mathsf{BUAF}_{\mathcal{P}} = \langle \mathbb{A}_{\mathcal{P}}, \mathbb{R}_{\mathcal{P}}, \mathbb{S}_{\mathcal{P}}, \preceq_{\mathcal{P}}\rangle$ the backing-undercutting argumentation framework associated with $\mathcal{P}$. For every pair of arguments $\mathcal{A}, \mathcal{B} \in \mathbb{A}_{\mathcal{P}}$ s.t. $\mathcal{A}$ defeats $\mathcal{B}$ in $\mathsf{BUAF}_{\mathcal{P}}$, $\mathcal{A}$ defeats $\mathcal{B}$ in $\mathcal{P}$.

To illustrate the occurrence of defeats in E-DeLP, let us consider the following example.

Example 12. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4 and $\mathsf{BUAF}_{\mathcal{P}_4}$ its associated backing-undercutting argumentation framework from Example 11. Then, we obtain the following defeats between arguments of $\mathcal{P}_4$:

- $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$ defeats $\langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle$ in $\mathcal{P}_4$ since $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$ primarily defeats $\langle \mathcal{A}_2, \sim illuminated\_room(r)\rangle$ in $\mathsf{BUAF}_{\mathcal{P}_4}$.
- $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ defeats $\langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l)\rangle_b$ in $\mathcal{P}_4$ since $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ implicitly defeats $\langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l)\rangle_b$ in $\mathsf{BUAF}_{\mathcal{P}_7}$.
- $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ defeats $\langle \mathcal{A}_3, light\_on(l)\rangle$ in $\mathcal{P}_4$ since $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ primarily (and indirectly) defeats $\langle \mathcal{A}_3, light\_on(l)\rangle$ in $\mathsf{BUAF}_{\mathcal{P}_4}$.
- $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ defeats $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$ in $\mathcal{P}_4$ since $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ indirectly defeats $\langle \mathcal{A}_1, illuminated\_room(r)\rangle$ in $\mathsf{BUAF}_{\mathcal{P}_4}$.

Fig. 4 illustrates these defeats using the notation of the BUAF introduced in Section 5.1, where $\mathcal{A} \longrightarrow \mathcal{B}$ denotes a defeat from $\mathcal{A}$ to $\mathcal{B}$. In addition, Fig. 4 adopts the simplified representation for arguments of $\mathcal{P}_4$ introduced in Example 11.

Now that we have characterized the notion of defeat in E-DeLP we can extend the visual representation introduced in Section 4 to consider the defeats between arguments and, following the notation adopted by the BUAF, defeats in E-DeLP will be denoted using solid arrows. Therefore, for instance, the defeat from an argument $\mathcal{A}$ to an argument $\mathcal{B}$ in E-DeLP will be noted as $\mathcal{A} \longrightarrow \mathcal{B}$. In addition, for those attacks that result in defeats between arguments in E-DeLP, the visual representation will only include the solid arrows corresponding to the defeats. That is, if $\mathcal{A}$ attacks $\mathcal{B}$ (noted as $\mathcal{A} \dashrightarrow \mathcal{B}$) and also defeats it (noted as $\mathcal{A} \longrightarrow \mathcal{B}$), the visual representation will only include the solid arrow from $\mathcal{A}$ to $\mathcal{B}$. To illustrate this, let us consider the following example.

Example 13. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4. The arguments built from $\mathcal{P}_4$ and the attacks between them are illustrated in Fig. 2, corresponding to Example 10. Given the defeats identified in Example 12, we obtain an extended visual representation of arguments, attacks and defeats from $\mathcal{P}_4$, illustrated in Fig. 5. Note that, by Definition 28, defeats between arguments in E-DeLP do not necessarily originate from attacks between arguments. Such is the case of the defeat from $\langle \mathcal{A}_5, light\_on(l) \prec switch\_on(l)\rangle_u$ to $\langle \mathcal{A}_4, light\_on(l) \prec switch\_on(l)\rangle_b$, which corresponds to an implicit defeat in $\mathsf{BUAF}_{\mathcal{P}_4}$ (*i.e.,* in the BUAF associated with $\mathcal{P}_4$).

6. Acceptability calculus

Having defined the infrastructure necessary for the construction of arguments, and having determined the defeats between them, we will now address the problem of deciding which arguments are accepted in E-DeLP. Starting from the BUAF associated with an E-DeLP program, we will determine the acceptability status of arguments built from that program. Specifically, the acceptability status of arguments in a program $\mathcal{P}$ will be determined by their corresponding acceptability status in $\mathsf{BUAF}_{\mathcal{P}}$. Then, the conclusions of the accepted arguments will determine the

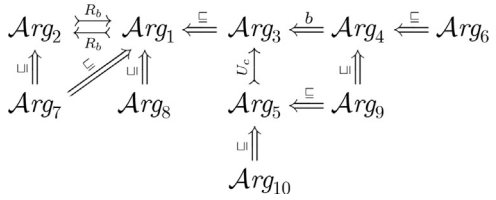

Fig. 3. Graphical Representation of $\mathcal{A}\mathsf{BUAF}_{\mathcal{P}_4}$ from Example 11.

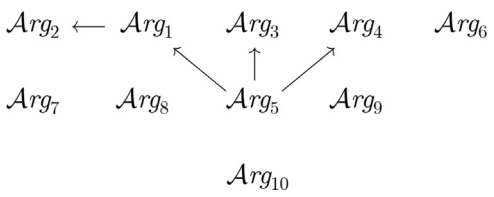

Fig. 4. Defeats between arguments of $\mathcal{P}_4$, corresponding to Example 12.

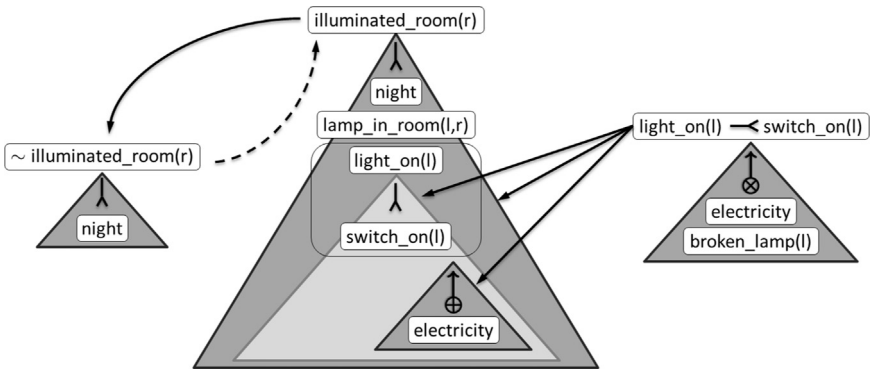

Fig. 5. Visual representation of arguments, attacks and defeats from $\mathcal{P}_4$, corresponding to Example 13.

inferences of the system, which will be referred to as the warranted literals from the corresponding E-DeLP program.

As mentioned in Section 5.1, in this work we will only consider the complete, preferred, stable, and grounded acceptability semantics. Then, as the following definition shows, an argument built from an E-DeLP program $\mathcal{P}$ will be accepted with respect to a given semantics iff it belongs to an extension of $\mathsf{BUAF}_{\mathcal{P}}$ under the same semantics.

Definition 29 (*E-DeLP extensions*). Let $\mathcal{P}$ be an extended defeasible logic program, $\mathsf{BUAF}_{\mathcal{P}}$ the backing-undercutting argumentation framework associated with $\mathcal{P}$, E a set of arguments built from $\mathcal{P}$ and $\mathsf{s} \in$ {complete, preferred, stable, grounded} an acceptability semantics. We say that E is an extension of $\mathcal{P}$ under the semantics s iff E is an extension of $\mathsf{BUAF}_{\mathcal{P}}$ under s.

Recall that, by Definition 24, the extensions of $\mathsf{BUAF}_{\mathcal{P}}$ are the extensions of its associated AF. Thus, the extensions of $\mathcal{P}$ are, in turn, the extensions of the AF associated with $\mathsf{BUAF}_{\mathcal{P}}$, which we may refer to as $\mathsf{AF}_{\mathcal{P}}$.

Example 14. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4 and $\mathsf{BUAF}_{\mathcal{P}_4}$ its associated backing-undercutting argumentation framework from Example 11. Example 12 shows the defeats between arguments of $\mathcal{P}_4$. Moreover, Fig. 4 depicts the arguments of $\mathcal{P}_4$ and the defeats between them, which characterize the $\mathsf{AF}_{\mathcal{P}_4}$ associated with $\mathsf{BUAF}_{\mathcal{P}_4}$. Thus, since there are no cycles of defeat in $\mathsf{AF}_{\mathcal{P}_4}$, the only complete, preferred, stable and grounded extension of $\mathcal{P}_4$ is

$$\mathsf{E}_4 = \left\{ \begin{array}{ll} \langle \mathcal{A}_2, \sim illuminated\_room(r) \rangle & \langle \emptyset, switch\_on(l) \rangle \\ \langle \mathcal{A}_5, light\_on(l) \text{–<} switch\_on(l) \rangle_u & \langle \emptyset, night \rangle \\ \langle \emptyset, lamp\_in\_room(l,r) \rangle & \langle \emptyset, electricity \rangle \\ \langle \emptyset, broken\_lamp(l) \rangle & \end{array} \right\} \quad \text{where}$$

$$\mathcal{A}_2 = \{ \sim illuminated\_room(r) \text{–<} night \}$$

$$\mathcal{A}_5 = \{ [light\_on(l) \text{–<} switch\_on(l)] \leftarrow\!\otimes [electricity, broken\_lamp(l)] \}$$

There is a vast amount of work in the literature showing that the application of alternative acceptability semantics may lead to obtaining different sets of extensions (see *e.g.* Dung, 1995; Baroni et al., 2011a). As the following example shows, there may exist arguments that do not belong to any extension of an E-DeLP program under a semantics s, arguments that belong to some extensions but not to others, and arguments that belong to every extension obtained under the semantics s.

Example 15. Let $\mathcal{P}_{15} = (\Pi_{15}, \Delta_{15}, \Sigma_{15})$ be an extended defeasible logic program, where $\Pi_{15} = \{a, b\}$, $\Delta_{15} = \{(l \text{–<} a), (\sim l \text{–<} b), (h \text{–<} a)\}$ and $\Sigma_{15} = \{[h \text{–<} a] \leftarrow\!\otimes [b]\}$. From $\mathcal{P}_{15}$ we obtain the following arguments:

$\langle \emptyset, a \rangle$ $\langle \mathcal{A}_1, l \rangle$ $\langle \mathcal{A}_3, h \rangle$
$\langle \emptyset, b \rangle$ $\langle \mathcal{A}_2, \sim l \rangle$ $\langle \mathcal{A}_4, h \text{–<} a \rangle_u$

with $\mathcal{A}_1 = \{l \text{–<} a\}$, $\mathcal{A}_2 = \{\sim l \text{–<} b\}$, $\mathcal{A}_3 = \{h \text{–<} a\}$ and $\mathcal{A}_4 = \{[h \text{–<} a] \leftarrow\!\otimes [b]\}$. In this case, $\langle \mathcal{A}_1, l \rangle$ rebuts $\langle \mathcal{A}_2, \sim l \rangle$ and vice-versa. In addition, $\langle \mathcal{A}_4, h \text{–<} a \rangle_u$ undercuts $\langle \mathcal{A}_3, h \rangle$. Suppose that the comparison criterion adopted in $\mathcal{P}_{15}$ is such that $\langle \mathcal{A}_1, l \rangle$ and $\langle \mathcal{A}_2, \sim l \rangle$ are equally preferred. Then, we obtain the following defeats between arguments in $\mathcal{P}_{15}$: on the one hand, $\langle \mathcal{A}_1, l \rangle$ and $\langle \mathcal{A}_2, \sim l \rangle$ defeat each other; on the other hand, since $\langle \mathcal{A}_3, h \rangle$ has no backing arguments, it is defeated by $\langle \mathcal{A}_4, h \text{–<} a \rangle_u$. Finally, considering the preferred semantics, we obtain the following preferred extensions of $\mathcal{P}_{15}$: $\mathsf{E}_{\mathcal{P}_{15}} = \{\langle \emptyset, a \rangle, \langle \emptyset, b \rangle, \langle \mathcal{A}_4, h \text{–<} a \rangle_u, \langle \mathcal{A}_1, l \rangle\}$ and $\mathsf{E}'_{\mathcal{P}_{15}} = \{\langle \emptyset, a \rangle, \langle \emptyset, b \rangle, \langle \mathcal{A}_4, h \text{–<} a \rangle_u, \langle \mathcal{A}_2, \sim l \rangle\}$.

As proposed in the literature (Touretzky et al., 1987), it is possible to characterize reasoners as skeptical or credulous depending on the approach taken for determining the acceptance or rejection of arguments. In particular, in this work we will follow a skeptical approach, requiring arguments to belong to every extension under a given semantics in order to be considered as accepted with respect to that semantics.

Definition 30 (*Accepted and rejected arguments*). Let $\mathcal{P}$ be an extended defeasible logic program, $\mathsf{s} \in$ {complete, preferred, stable, grounded} an acceptability semantics, and $\mathcal{A}$ an argument built from $\mathcal{P}$. If $\mathcal{A}$ belongs to every extension of $\mathcal{P}$ under the semantics s, then we will say that $\mathcal{A}$ is *accepted* with respect to s; otherwise, we will say that $\mathcal{A}$ is *rejected* with respect to s.

Example 16. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4. As shown in Example 14, $\mathcal{P}_4$ has only one complete, preferred, stable, and grounded extension. Therefore, all arguments belonging to the extension are accepted with respect to these semantics, whereas any other argument is rejected with respect to them.

Example 17. Given the extensions of the E-DeLP program $\mathcal{P}_{15}$ from Example 15, it holds that arguments $\langle \emptyset, a \rangle$, $\langle \emptyset, b \rangle$ and $\langle \mathcal{A}_4, h \text{–<} a \rangle_u$ are accepted with respect to the preferred semantics. In contrast, arguments $\langle \mathcal{A}_1, l \rangle$, $\langle \mathcal{A}_2, \sim l \rangle$ and $\langle \mathcal{A}_3, h \rangle$ are rejected with respect to that semantics. On the one hand, $\langle \mathcal{A}_1, l \rangle$ and $\langle \mathcal{A}_2, \sim l \rangle$ are rejected because they only belong to one preferred extension of $\mathcal{P}_{15}$. On the other hand, $\langle \mathcal{A}_3, h \rangle$ is rejected because it does not belong to any preferred extension of $\mathcal{P}_{15}$.

Having identified the accepted arguments of an E-DeLP program $\mathcal{P}$ under a semantics s, it is possible to determine the inferences of $\mathcal{P}$, which will be referred to as the *warranted literals* under the corresponding semantics. Intuitively, since claim arguments are the only ones whose conclusions are literals, the warranted literals from

an E-DeLP program will be the conclusions of all accepted claim arguments.

Definition 31 (*Warranted literals*). Let $\mathcal{P}$ be an extended defeasible logic program, L a literal and $s \in$ {complete, preferred, stable, grounded} an acceptability semantics. We say that L is *warranted* from $\mathcal{P}$ under the semantics s iff there exists a claim argument $\langle \mathcal{A}, L \rangle$ such that it is accepted with respect to the semantics s.

Example 18. Let $\mathcal{P}_4$ be the extended defeasible logic program from Example 4. As shown in Examples 14 and 16, $\mathcal{P}_4$ has only one complete, preferred, stable and grounded extension and thus, all arguments belonging to the extension are accepted with respect to those semantics. Hence, the warranted literals from $\mathcal{P}_4$ under the complete, preferred, stable and grounded semantics are "$\sim$*illuminated_room*(r)", "*lamp_in_room*(l, r)", "*broken_lamp*(l)", "*switch_on*(l)", "*electricity*" and "*night*".

Next, we will present a series of properties satisfied by E-DeLP, which refer to desirable features of argumentation systems. In particular, these properties concern the acceptance of arguments and the inferences of the system. First, we will show that all arguments built from the strict knowledge of an E-DeLP program will be accepted and, therefore, their conclusions will be warranted literals. Then, we will show that no pair of accepted arguments will be such that one defeats the other. Finally, we will show that the inferences obtained from an E-DeLP program are consistent since literals in disagreement cannot be simultaneously warranted.

The following proposition shows that, given an E-DeLP program $\mathcal{P}$, every argument whose conclusion has a strict derivation from $\mathcal{P}$ will be accepted. Moreover, the conclusions of those arguments will be warranted literals from the E-DeLP program.

Proposition 5. *Let $\mathcal{P}$ be an extended defeasible logic program and* $s \in$ {*complete*, *preferred*, *stable*, *grounded*} *an acceptability semantics. If $\langle \emptyset, L \rangle$ is an argument built from $\mathcal{P}$, then $\langle \emptyset, L \rangle$ is accepted with respect to the semantics* s *and L is a warranted literal from $\mathcal{P}$ under* s.

Proof. See Appendix.

Another property satisfied by E-DeLP regards the conflict-freeness of the sets of accepted arguments. Specifically, all extensions of an E-DeLP program will be such that they do not simultaneously accept arguments that defeat one another.

Proposition 6. *Let $\mathcal{P}$ be an extended defeasible logic program and* $s \in$ {*complete*, *preferred*, *stable*, *grounded*} *an acceptability semantics. If $\langle \mathcal{A}, H \rangle$ is an argument of $\mathcal{P}$ that is accepted with respect to the semantics* s, *then for every argument $\langle \mathcal{B}, Q \rangle$ s.t. $\langle \mathcal{A}, H \rangle$ defeats $\langle \mathcal{B}, Q \rangle$ or $\langle \mathcal{B}, Q \rangle$ defeats $\langle \mathcal{A}, H \rangle$ in $\mathcal{P}$, it holds that $\langle \mathcal{B}, Q \rangle$ is a rejected argument with respect to the semantics* s.

Proof. See Appendix.

The following theorem formalizes a fundamental property satisfied by E-DeLP, which regards the inferences of the system. Specifically, it shows that the inferences of an E-DeLP program are consistent, in the sense that the program does not warrant literals in disagreement. It should be noted that, although Proposition 6 shows a very simple result, it is useful for proving the following theorem.

Theorem 1. *Let $\mathcal{P}$ be an extended defeasible logic program and* $s \in$ {*complete*, *preferred*, *stable*, *grounded*} *an acceptability semantics. If L is a warranted literal from $\mathcal{P}$ under the semantics* s, *then for every literal L' s.t. L and L' disagree it holds that L' is not a warranted literal from $\mathcal{P}$ under the semantics* s.

Proof. See Appendix.

The property formalized by the preceding theorem relies on the fact that Definition 30 adopts a skeptical approach for characterizing the accepted arguments in E-DeLP (thus justifying the choice of skeptical over credulous acceptance). In contrast, if Definition 30 had adopted a credulous approach by requiring arguments to belong to *at least one* extension under a given semantics in order to be considered as accepted with respect to that semantics, Theorem 1 would no longer hold. To illustrate this, let us consider the E-DeLP program $\mathcal{P}_{15}$ from Example 15. By considering a credulous approach, arguments $\langle \mathcal{A}_1, l \rangle$ and $\langle \mathcal{A}_2, \sim l \rangle$ would be accepted with respect to the preferred semantics since they respectively belong to the preferred extensions $\mathsf{E}_{\mathcal{P}_{15}}$ and $\mathsf{E}'_{\mathcal{P}_{15}}$. Thus, by Definition 31, it would be the case that both literals "l" and "$\sim l$" are warranted from $\mathcal{P}_{15}$ under the preferred semantics. As a result, if we adopted a credulous approach, the inferences from the E-DeLP program $\mathcal{P}_{15}$ would no longer be consistent since the program would be simultaneously warranting literals in disagreement.

Finally, it is important to remark that Theorem 1 relates to the rationality postulate of *direct consistency* proposed in Caminada and Amgoud (2007). Specifically, the direct consistency postulate expresses that the set of justified conclusions of a structured argumentation system and the different sets of conclusions corresponding to each extension should be consistent. On the one hand, it is easy to see that the first part of this postulate is satisfied in E-DeLP since, by Theorem 1, literals in disagreement will not be simultaneously warranted from an E-DeLP program. On the other hand, since the extensions of an E-DeLP program $\mathcal{P}$ coincide with the extensions of $\mathsf{AF}_{\mathcal{P}}$ (the AF associated with $\mathsf{BUAF}_{\mathcal{P}}$, the backing-undercutting argumentation framework associated with $\mathcal{P}$) and, by Definition 26, these extensions are conflict-free, the second part of this postulate also holds.

7. Related work and discussion

We have introduced a structured argumentation system that extends the representational formalism of Defeasible Logic Programming (DeLP) (García and Simari, 2004). The system proposed here, called *Extended Defeasible Logic Programming* (E-DeLP), allows for the representation of attack and support for defeasible rules through the addition of backing and undercutting rules; in this way, the language of E-DeLP models the notion of backing proposed by Toulmin (1958), as well as the notion of undercut introduced by Pollock (1987). In addition, since E-DeLP keeps the elements of DeLP's representational language, the formalism proposed in this paper effectively extends that of García and Simari (2004), providing a way to apply any acceptability semantics in this framework.

In this work we follow the specification of E-DeLP programs proposed in Cohen et al. (2011), as well as the formalization of different types of arguments and the attacks between them. In particular, this is a further development of Cohen et al. (2011), where different aspects of E-DeLP are explored in more detail and others are corrected. Also, we have incorporated more examples and discussed the intuitions behind the formalization of the system. Another difference between this work and the one reported in Cohen et al. (2011) is that here we explicitly address the acceptability calculus of arguments in E-DeLP. In that way, we are able to identify the warranted literals from an E-DeLP program, which constitute the inferences of the system. Then, we also

formalize a series of properties satisfied by E-DeLP, which characterize desirable features of a structured argumentation system.

In order to determine the acceptance or rejection of arguments built from an E-DeLP program, we make use of the Backing-Undercutting Argumentation Framework (BUAF) proposed in Cohen et al. (2012). The BUAF adopts the same interpretation of support as E-DeLP, distinguishing between the support provided by sub-arguments and backings. The main difference between the BUAF and E-DeLP relies on their level of abstraction: whereas the former is an abstract argumentation framework, the latter is a structured argumentation system. Therefore, by characterizing the BUAF associated with an E-DeLP program, we show that E-DeLP provides the means for instantiating the formalism proposed in Cohen et al. (2012).

In the last decade, the study of the notion of support regained attention amongst researchers. In particular, Verheij was one of the first to resume the study of support, by performing a reconstruction of Toulmin's ideas (Verheij, 2005, 2009) and allowing for the representation of the elements in Toulmin's scheme and the support links between them. Verheij's proposal was developed using DefLog (Verheij, 2003), a theory of dialectical argumentation based on sentences, instead of based on arguments. This is because DefLog focuses on the justification of *prima facie* assumptions, instead of focusing on the arguments obtained in terms of them.

Briefly, DefLog's logical language has two connectives: $\times$ (dialectical negation) and $\rightsquigarrow$ (primitive implication). The dialectical negation $\times S$ of a sentence S expresses that S is defeated. Thus, the dialectical negation is inherently "directed" in the following sense: if $\times S$ is justified, then S is defeated; however, it is not always the case that if S is justified, then $\times S$ is defeated. On the other hand, the primitive implication $\rightsquigarrow$, in contrast to the material implication of classical logic, expresses elementary conditional relations that exist contingently in the world. It is a binary connective used to express that one sentence supports another, allowing one to obtain new sentences through the use of *modus ponens*. Finally, in DefLog it is possible to combine and nest the connectives $\times$ and $\rightsquigarrow$ to obtain more complex sentences like $A \rightsquigarrow (B \rightsquigarrow C)$ or $A \rightsquigarrow \times B$.

A main difference between DefLog and E-DeLP is that the former is a formalism based on sentences, whereas the latter is an argumentation system based on logic programming. Hence, arguments in DefLog are sets of sentences, whereas in E-DeLP arguments correspond to specific sets of rules (defeasible rules and/or backing rules). As mentioned before, in DefLog it is possible to combine and nest the connectives $\times$ and $\rightsquigarrow$, thus allowing for the representation of Toulmin's backings and Pollock's undercutting defeaters. Notwithstanding this, given that the use of dialectical negation implies the existence of defeats, an argument for a sentence $\times S$ will always be preferred to an argument for a sentence S. Therefore, in DefLog it is not possible to express attack without defeat; that is, the notions of attack and defeat are equivalent in Verheij's approach. In contrast, in E-DeLP we identify different kinds of attack between arguments (namely, rebutting, undercutting, and undermining). Then, by characterizing the BUAF associated with an E-DeLP program, we obtain the defeats between those arguments. Thus, in E-DeLP it is possible to represent both notions of attack and defeat, since the existence of attacks between arguments do not necessarily lead to the existence of defeats. Moreover, as shown in Section 5, an implicit defeat from an argument $\mathcal{A}$ to an argument $\mathcal{B}$ is not directly obtained from an attack between $\mathcal{A}$ and $\mathcal{B}$. Rather, the implicit defeat from $\mathcal{A}$ to $\mathcal{B}$ is obtained by combining other attacks and supports involving these arguments.

The authors of Amgoud et al. (2004) and Cayrol and Lagasquie-Schiex (2005) were the first researchers to explicitly account for a support relation in the context of abstract argumentation. They proposed the Bipolar Argumentation Framework (BAF), an abstract argumentation system incorporating a *general support* relation between arguments. In that way, the support relation is simply considered as a positive interaction, without imposing additional constraints. Then, the semantics associated with the support relation are given implicitly, by characterizing a series of *complex defeats* that arise from the combination of the original defeat and support relations.

The extensive research line on Bipolar Argumentation Frameworks motivated later studies on the notion of support. As a result, different formalizations of support relations were proposed in the literature, considering alternative interpretations for this notion. The most distinguished are the *evidential support* proposed in Oren and Norman (2008), the *deductive support* of Boella et al. (2010) and the *necessary support* of Nouioua and Risch (2011), which are surveyed in Cayrol and Lagasquie-Schiex (2013) and Cohen et al. (2014). Briefly, the notion of evidential support enables one to distinguish between *prima facie* and standard arguments. *Prima facie* arguments represent the notion of evidence and do not require support from other arguments to stand, while standard arguments cannot be accepted if they are not supported by evidence. On the other hand, deductive support is intended to capture the following intuition: if argument $\mathcal{A}$ supports argument $\mathcal{B}$, then the acceptance of $\mathcal{A}$ implies the acceptance of $\mathcal{B}$ and, conversely, the non-acceptance of $\mathcal{B}$ implies the non-acceptance of $\mathcal{A}$. Finally, the notion of necessary support enforces the following constraint: if argument $\mathcal{A}$ supports argument $\mathcal{B}$, then it means that $\mathcal{A}$ is necessary for $\mathcal{B}$. Thus, the acceptance of $\mathcal{B}$ implies the acceptance of $\mathcal{A}$ and, conversely, the non-acceptance of $\mathcal{A}$ implies the non-acceptance of $\mathcal{B}$.

Similar to the BAF, each of the formalizations mentioned above characterize a series of *complex defeats* that reinforce the acceptability constraints imposed by their corresponding interpretation of support. Moreover, it can be noted that there exists a relation between some of these interpretations. In particular, in Cayrol and Lagasquie-Schiex (2013) and Cohen et al. (2014), the notions of deductive support and necessary support are shown to be dual in the following sense: an argument $\mathcal{A}$ deductively supports an argument $\mathcal{B}$ iff $\mathcal{B}$ is necessary for $\mathcal{A}$. In addition, in Cayrol and Lagasquie-Schiex (2013) the authors state that evidential support cannot be reduced to necessary support (nor to deductive support). So, it is not possible to handle together in the same BAF the notions of evidential support and necessary/deductive support.

Notwithstanding this, in Polberg and Oren (2014) the authors present an improved characterization of the Evidential Argumentation System (EAS) proposed in Oren and Norman (2008), and they show that there exists a relation between the notions of evidential support and necessary support. In order to do this, they propose a translation from the Argumentation Framework with Necessities (AFN) of Nouioua and Risch (2011) into an EAS, and they also show how to translate the evidential support relation of the EAS into the necessary support relation of the AFN. In particular, the results presented in Polberg and Oren (2014) make use of the formalization of the AFN provided in Nouioua (2013), in which the necessity support relation is generalized in a way such that it originates from sets of arguments.

The most significant difference between E-DeLP and the formalizations of support presented above is that they correspond to abstract argumentation frameworks, whereas E-DeLP is a structured argumentation system based on logic programming. Taking this into account, we cannot perform a direct comparison between E-DeLP and the other formalisms; however, it is possible to identify a relation between the interpretations of support adopted by them. On the one hand, following the compositionality principle of Prakken and Vreeswijk (2002) which captures the intuition that an argument cannot be accepted unless all its sub-arguments are accepted, sub-arguments in E-DeLP can be considered as *necessary* for their super-arguments. Then, since backing arguments in E-DeLP are a special case of sub-

arguments, backing arguments can also be considered as *necessary* for the arguments they support.

On the other hand, it could be argued that there exists a relation between the notions of backing and evidential support. Recall that the notion of derivation in E-DeLP establishes that, if backing rules for a given defeasible rule exist, they must be taken into consideration in the derivation process. This may suggest that backing rules in E-DeLP provide some sort of evidence required for using their associated defeasible rules. However, there exists a fundamental difference between these two notions: arguments in an EAS cannot be accepted if they are not supported by evidence, whereas arguments in E-DeLP can be accepted even though they do not make use of backing rules (*i.e.,* even though there are no backing arguments supporting them). This is because the set of arguments of an EAS corresponds to *potential* arguments. Then, only those arguments that are supported by evidence are considered in the acceptability calculus. In contrast, since the existence of backing rules in E-DeLP is not mandatory, E-DeLP programs can be specified in a way such that they do not contain backing rules. Hence, in such a case, arguments built from that program will not contain backing rules and, consequently, will not have backing arguments.

Nonetheless, the notion of evidential support can be related to the existence of facts and presumptions in E-DeLP. Recall that facts correspond to information perceived as secure and indisputable, which may be the result of observations from the environment; thus, they can be considered as evidence. Then, since presumptions can only be used if they do not contradict the strict knowledge of an E-DeLP program, they can be considered as supported by the absence of evidence contradicting them. As a result, since facts and presumptions provide the grounds for every derivation leading to the construction of arguments, every argument built from an E-DeLP program will be directly or indirectly supported by evidence.

In Prakken (2009) the author introduced ASPIC+, an argumentation formalism that instantiates Dung (1995)'s approach by partially specifying the structure of arguments and identifying different kinds of attack between them. One difference between ASPIC+ and E-DeLP is that the former proposes a general notion of argument, whereas the latter distinguishes between three types of arguments, including backing and undercutting arguments.

Similar to ASPIC+, E-DeLP considers rebutting attacks, undercutting attacks and undermining attacks; however, unlike ASPIC+, attacks in E-DeLP are not resolved directly by applying preferences. Instead, as mentioned before, we characterize a BUAF associated with an E-DeLP program that allows one to obtain the defeats between arguments. Moreover, there is an essential difference in the resolution of undercutting attacks in ASPIC+ and in E-DeLP. Undercutting attacks in ASPIC+ will always succeed. In contrast, the resolution of undercutting attacks in the BUAF associated with an E-DeLP program requires the consideration of backing arguments that support the attacked argument; this highlights another difference between E-DeLP and ASPIC+. In addition to the possibility of expressing reasons against defeasible inference rules (through the use of undercutting rules), E-DeLP allows for the consideration of reasons in favor of using defeasible rules, determining conditions under which they are applicable. Thus, E-DeLP is the first structured argumentation system to address Toulmin's and Pollock's ideas simultaneously.

Finally, our work relates to Prakken (2014) in that they both consider the instantiation of abstract argumentation formalisms with more concrete ones. On the one hand, Prakken (2014) is concerned with the analysis of whether the BAF (Cayrol and Lagasquie-Schiex, 2005), the EAS (Oren and Norman, 2008), and the SuppAF framework proposed in Prakken (2014) can be instantiated by considering ASPIC+'s sub-argument relation. On the other hand, in this paper we have shown that, given the specification of an E-DeLP program, it is possible to provide a characterization of its associated BUAF (Cohen et al., 2012). As mentioned before, this is performed with the aim of addressing the acceptability calculus in E-DeLP. However, this also shows that there exists a correspondence between the support relations of E-DeLP and the corresponding BUAF, modeling the support that Toulmin's backings provide for their associated warrants in different contexts. Whereas E-DeLP provides a characterization of backings in a structured argumentation system, the BUAF models this notion in the context of abstract argumentation.

8. Conclusions and future work

In this work we proposed a structured argumentation system called *Extended Defeasible Logic Programming* (E-DeLP) inspired by the ideas of Toulmin (1958) and Pollock (1987). As shown in Section 2, E-DeLP extends the formalism of García and Simari (2004) by incorporating two new kinds of rules that allow one to express reasons for and against the use of defeasible rules. Then, in Section 3 we introduced the different types of arguments that can be built from an E-DeLP program: claim arguments, backing arguments and undercutting arguments, where the last two are closely related to Toulmin's notion of backing and Pollock's notion of undercut.

Given the arguments built from an E-DeLP program, in Section 4 we identified three forms of attack that can occur between them: rebutting, undercutting, and undermining attacks. On the one hand, the first and third forms of attack dispute the conclusion of an argument. On the other hand, the second form of attack provides a reason against a defeasible rule used by another argument. It can be noted that backing and undercutting arguments have an opposing nature. Whereas undercutting arguments originate undercutting attacks, backing arguments are aimed at avoiding the success of those attacks by providing a defense for the defeasible rules in dispute.

Following the query-oriented approach of García and Simari (2004), in E-DeLP we are interested in determining the inferences of the system, which correspond to the warranted literals from an E-DeLP program. To achieve this, in Section 5 we introduced the characterization of a BUAF associated with an E-DeLP program, from which we obtained the defeats between arguments. Then, by considering the arguments built from an E-DeLP program and the defeats between them, in Section 6 we addressed the acceptability calculus of arguments by adopting an extensional approach. Specifically, the accepted arguments of an E-DeLP program were identified in terms of the extensions of its associated BUAF. From the accepted arguments we distinguished the claim arguments, which determined the inferences of the system. As a consequence, the conclusions of all claim arguments built from an E-DeLP program correspond to the warranted literals from that program. Finally, in Section 6 we also studied several properties satisfied by E-DeLP, regarding the acceptance of arguments and the consistency of the system's inferences.

The approach taken in Section 6 has two clear advantages. First, by defining the extensions of an E-DeLP program in terms of the extensions of its associated BUAF (which, in turn, are obtained in terms of its associated AF), we are able to use any acceptability semantics for AF's proposed in the literature (see Baroni and Giacomin, 2009; Baroni et al., 2011a for an overview on argumentation semantics). Second, by providing a characterization of the associated BUAF, we showed that it is possible to instantiate the formalism of Cohen et al. (2012) starting from the specification of an E-DeLP program. In that way, we provided a concrete application of the abstract argumentation framework proposed in Cohen et al. (2012).

Similar to E-DeLP, there exist other approaches in the literature that address the existence of support between arguments; however, as mentioned in Section 7, they are mostly developed in a context of abstract argumentation. As a result, E-DeLP is the first argumentation system to combine Toulmin's and Pollock's ideas in a context of structured argumentation. Moreover, E-DeLP is the first structured argumentation system enabling one to argue about the use of inference rules. In particular, as mentioned before, in DeLP García and Simari (2004) it is not possible to argue about that since only literals can appear in the head of rules; thus, defeasible rules in DeLP cannot be attacked or supported. As a result, by incorporating backing and undercutting rules, E-DeLP enriches the representational capabilities of DeLP, resulting in a more expressive logic programming language and thus, allowing for its use in more complex scenarios.

Given that backing and undercutting rules in E-DeLP provide (defeasible) reasons supporting or attacking the use of defeasible rules, they could also be considered as defeasible. Therefore, it should be possible to provide reasons for or against their use. The aim of this work was to provide a formalization of Toulmin's and Pollock's ideas in the context of a structured argumentation system. Hence, the representational language of E-DeLP proposed in this work was kept as simple as possible and thus, it does not currently allow for attack or support on backing/undercutting rules. Notwithstanding this, motivated on the work by Modgil (2009) and Baroni et al. (2011b), where attacks on attacks are allowed in abstract argumentation frameworks, we started to study the possibility of modeling recursive attack and support relations (as well as their combination) in the context of abstract argumentation (Cohen et al., 2014b). Then, following that work, we are interested in further studying Pollock's and Toulmin's ideas in the context of structured argumentation, by analyzing the possibility of expressing attack and support for inference rules in a recursive manner. That is, to allow for backing and undercutting rules to be themselves attacked and supported.

The formalization of E-DeLP as a structured argumentation system based on logic programming facilitates its computational implementation. In particular, we are currently starting to work on a full implementation of E-DeLP, by extending the existing implementation of DeLP García (2000) and García et al. (2007). First, we have to extend the internal representation of rules, in order to give the possibility of expressing reasons for and against using defeasible rules. For that purpose, we are planning on associating labels to defeasible rules, which will be treated as special literals. Hence, the reasons for or against the use of a given defeasible rule will be specified by referring to the label associated with that rule or its complement. Moreover, we will have to extend the existing implementation of DeLP's derivation mechanism in order to account for backing rules and their relevance on the derivation process in E-DeLP. After building the arguments and identifying the attacks and supports between them, we will implement the mapping leading to obtain the BUAF associated with an E-DeLP program. Then, the mechanism for obtaining the defeats between arguments can be easily implemented and thus, we will specify the AF associated with the BUAF. Finally, in order to obtain the accepted arguments, we will make use of the implementations of argumentation semantics available in the literature (Nofal et al., 2014).

To conclude, let us consider the use of DeLP in concrete domains and real-world applications. In Ferretti et al. (2008) the authors present a model for defeasible decision making using DeLP. The principles stated in their work are exemplified in a robotic domain where a *Khepera* 2 robot collects boxes scattered through the environment and therefore, should make decisions about which box must be transported next. Another concrete application of DeLP can be found in Gómez et al. (2013), where the authors propose ONTOarg, a decision support framework for performing local-as-view integration of possibly inconsistent and incomplete ontologies in terms of DeLP. Finally, in Deagustini et al. (2013) the authors deal with the problem of massively built arguments from premises obtained from relational databases and compute warrant. There, the authors propose and test a series of algorithms for integrating a database management system with DeLP's argument-based inference engine.

By expanding the formalism of García and Simari (2004) and providing an implementation of E-DeLP we will be improving the knowledge representation and reasoning capabilities of DeLP. As a result, current users of DeLP will be provided with more expressive tools to work on their application domains. Moreover, by allowing one to argue about the use of inference rules, E-DeLP will be suitable for application on more complex domains where DeLP could not be used. For instance, this provides an interesting possibility for developing new architectures for knowledge-based applications, such as Decision Support Systems and Recommender Systems that could efficiently use argumentation as the underlying inference model.

Acknowledgments

We would like to thank the anonymous reviewers for their helpful comments on the previous versions of this paper. This work was partially supported by PGI-UNS (Grants 24/N035, 24/N030) and PIP-CONICET (Grant 112-201101-01000).

Appendix A

This appendix includes the proofs of the formal results introduced in the paper.

Proposition 1. *Let $\mathcal{P}=(\Pi,\Delta,\Sigma)$ be an extended defeasible logic program and $\langle\mathcal{A},H\rangle$ a claim argument built from $\mathcal{P}$. If R is a top defeasible rule of $\langle\mathcal{A},H\rangle$, then $\nexists R'\in\Delta$ such that $R'\neq R$ and R' is a top defeasible rule of $\langle\mathcal{A},H\rangle$.*

Proof. Given the claim argument $\langle\mathcal{A},H\rangle$, let us suppose by contradiction that $\exists R,R'\in\Delta$ s.t. $R'\neq R$ and both R and R' are top defeasible rules of $\langle\mathcal{A},H\rangle$. Then, by Definition 11, R: "H–≺*Body*", R': "H–≺*Body*′", $R\in\mathcal{A}$ and $R'\in\mathcal{A}$. That is, $\langle\mathcal{A},H\rangle$ contains two different defeasible rules for the same conclusion. Then, $\langle\mathcal{A},H\rangle$ would not be minimal and thus, by Definition 10, would not be a claim argument built from $\mathcal{P}$, which contradicts the hypothesis.□

Proposition 2. *Let $\mathcal{P}$ be an extended defeasible logic program and $\langle\mathcal{A},H\rangle$ an argument built from $\mathcal{P}$. If $\exists BR\in\mathcal{A}$ such that BR is a backing rule for a defeasible rule $R\in\mathcal{A}$, then there exists a backing argument $\langle\mathcal{B},R\rangle_b$ such that $BR\in\mathcal{B}$ and $\langle\mathcal{B},R\rangle_b$ is a sub-argument of $\langle\mathcal{A},H\rangle$.*

Proof. If $\langle\mathcal{A},H\rangle$ is an argument built from $\mathcal{P}$ and there exists a backing rule $BR\in\mathcal{A}$ for a defeasible rule $R\in\mathcal{A}$, then, by Definitions 8, 10, 12 and 13, there exists $\mathcal{S}\subseteq\mathcal{A}$ such that for every literal in BR's body there is a derivation from $\mathcal{S}\cup\Pi$. Also, by Definition 9, $\mathcal{S}$ is minimal and $\mathcal{S}\cup\Pi$ is non-contradictory. Thus, by Definition 12, there exists a backing argument $\langle\mathcal{B},R\rangle_b$, with $\mathcal{B}=\mathcal{S}\cup\{BR\}$. Furthermore, since $\mathcal{B}\subseteq\mathcal{A}$, $\langle\mathcal{B},R\rangle_b$ is a sub-argument of $\langle\mathcal{A},H\rangle$.□

Proposition 3. *Let $\mathcal{P}$ be an extended defeasible logic program and $\langle\mathcal{A},H\rangle$ an argument built from $\mathcal{P}$. Every proper sub-argument of $\langle\mathcal{A},H\rangle$ is either a claim argument or a backing argument.*

Proof. Let $\mathcal{P}=(\Pi,\Delta,\Sigma)$. We will consider the following alternatives: (a) $\langle\mathcal{A},H\rangle$ is a claim argument, (b) $\langle\mathcal{A},H\rangle$ is a backing argument, or (c) $\langle\mathcal{A},H\rangle$ is an undercutting argument.

(a) If $\langle\mathcal{A},H\rangle$ is a claim argument, then, by Definition 10, there exists a derivation for H from $\Pi\cup\mathcal{A}$. Also, by Definition 8, the

derivation of H does not make use of undercutting rules. Then, by condition (4) in Definition 10, $\langle \mathcal{A}, H\rangle$ is minimal and thus, $\mathcal{A}$ does not contain undercutting rules. Therefore, no proper subset $\mathcal{B}$ of $\mathcal{A}$ will be such that it contains undercutting rules. As a result, every proper sub-argument $\langle \mathcal{B}, Q\rangle$ of $\langle \mathcal{A}, H\rangle$ will be a claim argument or a backing argument.

(b) If $\langle \mathcal{A}, H\rangle$ is a backing argument, then, by Definition 12, $\mathcal{A} = \{[H] \leftarrow \oplus [L_1, \ldots, L_n]\} \cup \mathcal{A}'$ and there exists a derivation for each L_i $(1 \leq i \leq n)$ from $\Pi \cup \mathcal{A}'$. By Definition 8, the derivation of each L_i does not make use of undercutting rules. Then, by condition (4) in Definition 12, $\langle \mathcal{A}, H\rangle$ is minimal and thus, $\mathcal{A}$ does not contain undercutting rules. Therefore, no proper subset $\mathcal{B}$ of $\mathcal{A}$ will be such that it contains undercutting rules. As a result, every proper sub-argument $\langle \mathcal{B}, Q\rangle$ of $\langle \mathcal{A}, H\rangle$ will be a claim argument or a backing argument.

(c) If $\langle \mathcal{A}, H\rangle$ is an undercutting argument, then, by Definition 13, $\mathcal{A} = \{[H] \leftarrow \otimes [L_1, \ldots, L_n]\} \cup \mathcal{A}'$ and there exists a derivation for each L_i $(1 \leq i \leq n)$ from $\Pi \cup \mathcal{A}'$. By Definition 8, the derivation of each L_i does not make use of undercutting rules. Then, by condition (4) in Definition 13, $\langle \mathcal{A}, H\rangle$ is minimal and thus, $\mathcal{A}$ contains exactly one undercutting rule: $[H] \leftarrow \otimes [L_1, \ldots, L_n]$. Suppose by contradiction that there exists $\mathcal{B} \subset \mathcal{A}$ s.t. $\langle \mathcal{B}, Q\rangle$ is an undercutting argument. If that were the case, then $\mathcal{B}$ must contain an undercutting rule. Then, since the only undercutting rule in $\mathcal{A}$ is $[H] \leftarrow \otimes [L_1, \ldots, L_n]$, it must be the case that $\langle \mathcal{B}, Q\rangle$ is an undercutting argument for the defeasible rule H (and thus, $Q = H$). As a result, there would exist a proper subset $\mathcal{B}$ of $\mathcal{A}$ that satisfies the conditions in Definition 13, contradicting the fact that $\langle \mathcal{A}, H\rangle$ is an undercutting argument for H (since $\mathcal{A}$ would not be minimal). Finally, for each proper sub-argument $\langle \mathcal{B}, Q\rangle$ of $\langle \mathcal{A}, H\rangle$, it must be the case that $\langle \mathcal{B}, Q\rangle$ is a claim argument or a backing argument. □

Proposition 4. *Let $\mathcal{P}$ be an extended defeasible logic program and $\mathsf{BUAF}_{\mathcal{P}} = \langle \mathbb{A}_{\mathcal{P}}, \mathbb{R}_{\mathcal{P}}, \mathbb{S}_{\mathcal{P}}, \preceq_{\mathcal{P}}\rangle$ the backing-undercutting argumentation framework associated with $\mathcal{P}$. For every $\langle \mathcal{A}, H\rangle \in \mathbb{A}_{\mathcal{P}}$ it holds that if $\exists \langle \mathcal{B}, R\rangle \in \mathbb{A}_{\mathcal{P}}$ such that $(\langle \mathcal{B}, R\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{B}_{k\mathcal{P}}$, then $\nexists \langle \mathcal{C}, R'\rangle \in \mathbb{A}_{\mathcal{P}}$ such that $\langle \mathcal{C}, R'\rangle \neq \langle \mathcal{B}, R\rangle$ and $(\langle \mathcal{C}, R'\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{B}_{k\mathcal{P}}$.*

Proof. Let $\langle \mathcal{A}, H\rangle, \langle \mathcal{B}, R\rangle, \langle \mathcal{C}, R'\rangle \in \mathbb{A}_{\mathcal{P}}$ be such that $(\langle \mathcal{B}, R\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{B}_{k\mathcal{P}}$ and $(\langle \mathcal{C}, R'\rangle, \langle \mathcal{A}, H\rangle) \in \mathbb{B}_{k\mathcal{P}}$. Then, by Definition 27, both R and R' are top defeasible rules of the claim argument $\langle \mathcal{A}, H\rangle$. However, by Proposition 1, claim arguments have only one top defeasible rule. Therefore, it must be the case that $R = R'$. On the other hand, by Definition 27, $\langle \mathcal{B}, R\rangle$ and $\langle \mathcal{C}, R'\rangle$ are proper sub-arguments of $\langle \mathcal{A}, R\rangle$. Finally, since $R = R'$ and, by Definition 10, $\langle \mathcal{A}, H\rangle$ is minimal, it must be the case that $\mathcal{B} = \mathcal{C}$ and thus, $\langle \mathcal{B}, R\rangle = \langle \mathcal{C}, R'\rangle$.□

Proposition 5. *Let $\mathcal{P}$ be an extended defeasible logic program and $\mathsf{s} \in \{$complete, preferred, stable, grounded$\}$ an acceptability semantics. If $\langle \emptyset, L\rangle$ is an argument built from $\mathcal{P}$, then $\langle \emptyset, L\rangle$ is accepted with respect to the semantics* s *and L is a warranted literal from $\mathcal{P}$ under* s.

Proof. Let $\mathcal{P} = (\Pi, \Delta, \Sigma)$. By Definition 10, $\langle \emptyset, L\rangle$ is a claim argument. Then, since $\langle \emptyset, L\rangle$ has an empty set of rules, there exists a strict derivation of L from Π. By Definitions 10, 12 and 13, for every argument $\langle \mathcal{A}, H\rangle$ built from $\mathcal{P}$ it holds that $\mathcal{A} \cup \Pi$ is a non-contradictory set. Hence, it is not possible to build an argument $\langle \mathcal{B}, Q\rangle$ from $\mathcal{P}$ s.t. the literals L and Q disagree. As a result, by Definitions 16 and 18, there will be no rebutting or undermining attacks on $\langle \emptyset, L\rangle$. On the other hand, since $\langle \emptyset, L\rangle$ does not make use of defeasible rules, by Definition 17, there will be no undercutting attacks on $\langle \emptyset, L\rangle$. Then, by Definitions 27 and 28, $\langle \emptyset, L\rangle$ will have no defeaters. Finally, by Definitions 24, 29 and 30, $\langle \emptyset, L\rangle$ will belong to every extension of $\mathcal{P}$ under the semantics s and thus, it will be accepted with respect to the semantics s. As a result, by Definition 31, L will be a warranted literal from $\mathcal{P}$ under s.□

Proposition 6. *Let $\mathcal{P}$ be an extended defeasible logic program and $\mathsf{s} \in \{$complete, preferred, stable, grounded$\}$ an acceptability semantics. If $\langle \mathcal{A}, H\rangle$ is an argument of $\mathcal{P}$ that is accepted with respect to the semantics* s, *then for every argument $\langle \mathcal{B}, Q\rangle$ s.t. $\langle \mathcal{A}, H\rangle$ defeats $\langle \mathcal{B}, Q\rangle$ or $\langle \mathcal{B}, Q\rangle$ defeats $\langle \mathcal{A}, H\rangle$ in $\mathcal{P}$, it holds that $\langle \mathcal{B}, Q\rangle$ is a rejected argument with respect to the semantics* s.

Proof. Direct from Definitions 29, 24, 26, 25, 28 and 30.□

Theorem 1. *Let $\mathcal{P}$ be an extended defeasible logic program and $\mathsf{s} \in \{$complete, preferred, stable, grounded$\}$ an acceptability semantics. If L is a warranted literal from $\mathcal{P}$ under the semantics* s, *then for every literal L' s.t. L and L' disagree it holds that L' is not a warranted literal from $\mathcal{P}$ under the semantics* s.

Proof. If L is a warranted literal from $\mathcal{P}$ under the semantics s, then, by Definition 31, there exists an argument $\langle \mathcal{A}, L\rangle$ that is accepted with respect to the semantics s. If there exists an argument $\langle \mathcal{B}, L'\rangle$ built from $\mathcal{P}$, then, by Definitions 16 and 18, it would be the case that $\langle \mathcal{A}, L\rangle$ rebuts or undermines $\langle \mathcal{B}, L'\rangle$ and vice-versa. Then, by Definitions 27 and 20, the backing-undercutting argumentation framework $\mathsf{BUAF}_{\mathcal{P}}$ associated with $\mathcal{P}$ will be such that either $\langle \mathcal{A}, L\rangle$ defeats $\langle \mathcal{B}, L'\rangle$, $\langle \mathcal{B}, L'\rangle$ defeats $\langle \mathcal{A}, L\rangle$, or $\langle \mathcal{A}, L\rangle$ defeats $\langle \mathcal{B}, L'\rangle$ and vice-versa. As a result, by Definition 28 and Proposition 6, it would be the case that $\langle \mathcal{B}, L'\rangle$ is rejected with respect to s and thus, by Definition 31, L' will not be a warranted literal from $\mathcal{P}$ under the semantics s.□

References

Alferes, J.J., Pereira, L.M., Przymusinski, T.C., 1996. Strong and explicit negation in non-monotonic reasoning and logic programming. In: JELIA 1996, pp. 143–163.

Amgoud, L., Cayrol, C., 2002. A reasoning model based on the production of acceptable arguments. Ann. Math. Artif. Intell. 34 (1–3), 197–215.

Amgoud, L., Cayrol, C., Lagasquie-Schiex, M.-C., 2004. On the bipolarity in argumentation frameworks. In: NMR 2004, pp. 1–9.

Amgoud, L., Maudet, N., Parsons, S., 2002. An argumentation-based semantics for agent communication languages. In: ECAI 2002, pp. 38–42.

Amgoud, L., Parsons, S., Maudet, N., 2000. Arguments, dialogue, and negotiation. In: ECAI 2000, pp. 338–342.

Amgoud, L., Vesic, S., 2011. A new approach for preference-based argumentation frameworks. Ann. Math. Artif. Intell. 63 (2), 149–183.

Baroni, P., Caminada, M., Giacomin, M., 2011a. An introduction to argumentation semantics. Knowl. Eng. Rev. 26 (4), 365–410.

Baroni, P., Cerutti, F., Giacomin, M., Guida, G., 2011b. AFRA: argumentation framework with recursive attacks. Int. J. Approx. Reason. 52 (1), 19–37.

Baroni, P., Giacomin, M., 2009. Semantics of abstract argument systems. In: Argumentation in Artificial Intelligence. Springer, Dordrecht, The Netherlands, pp. 25–44.

Bench-Capon, T.J.M., Dunne, P.E., 2007. Argumentation in artificial intelligence. Artif. Intell. 171 (10–15), 619–641.

Besnard, P., Hunter, A., 2008. Elements of Argumentation. MIT Press, London, UK.

Black, E., Hunter, A., 2009. An inquiry dialogue system. Auton. Agents Multi-Agent Syst. 19 (2), 173–209.

Boella, G., Gabbay, D.M., van der Torre, L.W.N., Villata, S., 2010. Support in abstract argumentation. In: COMMA 2010. Frontiers in Artificial Intelligence and Applications, vol. 216, pp. 111–122.

Caminada, M., 2006. Semi-stable semantics. In: COMMA 2006, pp. 121–130.

Caminada, M., Amgoud, L., 2007. On the evaluation of argumentation formalisms. Artif. Intell. 171 (5–6), 286–310.

Cayrol, C., Lagasquie-Schiex, M.-C., 2005. On the acceptability of arguments in bipolar argumentation frameworks. In: ECSQARU 2005, pp. 378–389.

Cayrol, C., Lagasquie-Schiex, M.-C., 2013. Bipolarity in argumentation graphs: Towards a better understanding. Int. J. Approx. Reason. 54 (7), 876–899.

Cohen, A., García, A.J., Simari, G.R., 2011. Backing and undercutting in defeasible logic programming. In: ECSQARU 2011, pp. 50–61.

Cohen, A., García, A.J., Simari, G.R., 2012. Backing and undercutting in abstract argumentation frameworks. In: FoIKS 2012, pp. 107–123.

Cohen, A., Gottifredi, S., García, A.J., Simari, G.R., 2014a. A survey of different approaches to support in argumentation systems. Knowl. Eng. Rev. 29 (5), 513–550.

Cohen, A., Gottifredi, S., García, A.J., Simari, G.R., 2014b. An approach to abstract argumentation with recursive attack and support. J. Appl. Log. http://dx.doi.org/10.1016/j.jal.2014.12.001

Deagustini, C.A.D., Fulladoza Dalibón, S.E., Gottifredi, S., Falappa, M.A., Chesñevar, C. I., Simari, G.R., 2013. Relational databases as a massive information source for defeasible argumentation. Knowl. Based Syst. 51, 93–109.
Dung, P.M., 1995. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. Artif. Intell. 77 (2), 321–358.
Dung, P.M., Mancarella, P., Toni, F., 2007. Computing ideal sceptical argumentation. Artif. Intell. 171 (10–15), 642–674.
Ferretti, E., Errecalde, M., García, A.J., Simari, G.R., 2008. Decision rules and arguments in defeasible decision making. In: COMMA 2008, pp. 171–182.
García, A.J., 2000. Defeasible Logic Programming: Definition, Operational Semantics and Parallelism (Ph.D. thesis). Universidad Nacional del Sur, Bahía Blanca, Argentina.
García, A.J., Rotstein, N.D., Tucat, M., Simari, G.R., 2007. An argumentative reasoning service for deliberative agents. In: KSEM 2007, pp. 128–139.
García, A.J., Simari, G.R., 2004. Defeasible logic programming: an argumentative approach. Theory Pract. Log. Program. 4 (1–2), 95–138.
Gómez, S.A., Chesnevar, C.I., Simari, G.R., 2013. ONTOarg: A decision support framework for ontology integration based on argumentation. Expert Syst. Appl. 40 (5), 1858–1870.
Lifschitz, V., 1996. Foundations of logic programs. In: Brewka, G. (Ed.), Principles of Knowledge Representation. CSLI Pub, Stanford, USA, pp. 69–128.
Modgil, S., 2009. Reasoning about preferences in argumentation frameworks. Artif. Intell. 173 (9–10), 901–934.
Nofal, S., Atkinson, K., Dunne, P.E., 2014. Algorithms for argumentation semantics: labeling attacks as a generalization of labeling arguments. J. Artif. Intell. Res. 49, 635–668.
Nouioua, F., 2013. Afs with necessities: further semantics and labelling characterization. In: SUM 2013, pp. 120–133.
Nouioua, F., Risch, V., 2011. Argumentation frameworks with necessities. In: SUM 2011, pp. 163–176.
Nute, D., 1988. Defeasible reasoning: a philosophical analysis in PROLOG. In: Fetzer, J. H. (Ed.), Aspects of Artificial Intelligence. Kluwer Academic Pub., Dordrecht, The Netherlands, pp. 251–288.
Oren, N., Norman, T.J., 2008. Semantics for evidence-based argumentation. In: COMMA 2008, pp. 276–284.
Parsons, S., Sierra, C., Jennings, N.R., 1998. Agents that reason and negotiate by arguing. J. Log. Comput. 8 (3), 261–292.
Polberg, S., Oren, N., 2014. Revisiting support in abstract argumentation systems. In: COMMA 2014, pp. 369–376.
Pollock, J.L., 1987. Defeasible reasoning. Cogn. Sci. 11 (4), 481–518.
Prakken, H., 2009. An abstract framework for argumentation with structured arguments. J. Argum. Comput. 1, 93–124.
Prakken, H., 2014. On support relations in abstract argumentation as abstractions of inferential relations. In: ECAI 2014, pp. 735–740.
Prakken, H., Sartor, G., 2002. The role of logic in computational models of legal argument: a critical survey. In: Computational Logic: Logic Programming and Beyond, pp. 342–381.
Prakken, H., Vreeswijk, G., 2002. Logics for defeasible argumentation. In: Gabbay, D., Guenthner, F. (Eds.), Handbook of Philosophical Logic, vol. 4. Kluwer Academic Pub., Dordrecht, The Netherlands, pp. 218–319.
Rahwan, I., Simari, G.R., 2009. Argumentation in Artificial Intelligence. Springer, Heidelberg, Germany.
Reed, C., Grasso, F., Rahwan, I., Simari, G.R. (Eds.), 2014. Argument & Computation—Special Issue: Tutorials on Structured Argumentation, vol. 5 (1). Taylor & Francis, London, UK.
Toulmin, S.E., 1958. The Uses of Argument. Cambridge University Press, Cambridge, UK.
Touretzky, D.S., Horty, J.F., Thomason, R.H., 1987. A clash of intuitions: The current state of nonmonotonic multiple inheritance systems. In: IJCAI 1987, pp. 476–482.
Verheij, B., 2003. DefLog: on the logical interpretation of prima facie justified assumptions. J. Log. Comput. 13 (3), 319–346.
Verheij, B., 2005. Evaluating arguments based on Toulmin's scheme. Argumentation 19 (3), 347–371.
Verheij, B., 2009. The Toulmin argument model in artificial intelligence. or: how semi-formal, defeasible argumentation schemes creep into logic. In: Rahwan, I., Simari, G.R. (Eds.), Argumentation in Artificial Intelligence. Springer, Dordrecht, The Netherlands, pp. 219–238 (Chapter 11).